PROXIMITY DETECTION WITH BLUETOOTH AND GPS USING ANDROID SMARTPHONES

BY

LIM YUEN NEE

A REPORT
SUBMITTED TO
Universiti Tunku Abdul Rahman
in partial fulfillment of the requirements
for the degree of
BACHELOR OF INFORMATION SYSTEM ENGINEERING (HONS)
Faculty of Information and Communication Technology
(Perak Campus)

MAY 2012

DECLARATION OF ORIGINALITY

I declare that this report entitled "**Proximity Detection with Bluetooth and GPS Using Android Smartphones**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature	:	____________________________
Name	:	__Lim Yuen Nee____________
Date	:	__13 August 2012___________

Acknowledgement

First of all I would like express my sincere thanks and appreciation towards my supervisor, Mr.Ooi Boon Yaik who has helped me a lot since I started my final year project. He had guided and motivated me by giving me ideas and ways to overcome the problems that I face while doing this project. It is my first time creating a mobile application for Android and without my supervisor’s help, I would not have succeed in preparing and getting my application to work successfully.

Besides that, I would also like to thank my partner, Lee Yen Nee who had also helped me in debugging and testing out the application. Finally, I would like to thank my parents for their love, support and continuous encouragement throughout the course.

Abstract

This project is to develop an application based on android which is used in helping users to search back on their lost belongings or someone who got lost. We have made a research on several similar android applications such as Bluetooth Finder, Baby Bluetooth Alarm and etc but there are none of those application that are suitable for our scenario. For example, neither of these applications is able to retrieve back the location of the lost device. This application will be adding on few extra functions such as reconnection, SMS triggering, and GeoLocation to ease user in searching back lost belongings. Since most devices nowadays have Bluetooth technology readily installed and fixed inside their mobile devices, we can take advantage of this technology. Bluetooth technology will enable multiple Bluetooth devices to connect to each other. Spiral model will be implemented in developing this project. This application main audience would be users who want to keep track on their belongings as they are moving around so that they would feel secure as this application would also be able to help users to search back their belongings as well.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

1.0 Introduction & Motivation

This application will enable the user to search for their lost devices or friends (with Bluetooth switch on in their mobile) easier. This project will be able to help user to recover lost objects as long as the Bluetooth devices are switched on and kept within the object itself.

For example, someone lost their luggage in the airport or the Bluetooth connection connecting the paired device have been out of range. He/she can straight use this application to search where is the location of the missing luggage by simply turning on the GPS mode in the application itself. It will determine the location of the lost luggage whether it's still nearby or someone has already taken it away. If the luggage is still within the same building, then he/she can walk around to search where the luggage is as long as the Bluetooth is turn on in the lost device as well. He/she can do so by sending out message together with the GPS location to their nearby friends to help search for the missing luggage.

Bluetooth technology (Bluetooth SIG) has been chosen in this project because Bluetooth is a low-cost, low power consumption and secured type of technology that are most suitable to be installed in mobile devices.

Bluetooth Technology is a wireless signal (Bluetooth SIG) that uses radio wave to communicate between devices that have hardware or "small computer chip" in the devices. It is originally designed to replace the cables. By using Bluetooth in our mobile devices, it allows us to share photos, music, videos and a lot more between two paired devices. Bluetooth technology is not only limited to mobile devices but it is used in most of our daily devices such as computer, wireless keyboard, wireless mouse, and etc to share information wirelessly. Bluetooth operate within the users' Personal Area Network (PAN) (Bluetooth SIG) at a distance up to 50 meters depending on which type of class radio that are being used. For mobile devices, it uses the class 2 radio which consists of a range up to 10 meters in distance.

For example, by using the Bluetooth technology in our mobile phones, it helps ease our lifestyles such as we can use our mobile Bluetooth to send out and receive pictures and videos easily from another mobile as long as the Bluetooth is switch on. Bluetooth can be used easily by anyone regardless of their age because Bluetooth technology is coded in such a way that allows the user to automatically detect any nearby devices as long as their Bluetooth is switch on. The user will just need to select which device they want to pair and then they can start sharing information instantly.

Bluetooth uses proximity detection (Machine Design) to detect nearby Bluetooth devices. By doing so, it allows the Bluetooth users to easily detect anyone with their mobile Bluetooth switch on as long as the Bluetooth setting is set to "discoverable" mode. Proximity detection will only work if the device is near enough for the Bluetooth to detect (should be within the Bluetooth detection range as well), otherwise no Bluetooth devices will be found.

1.1 Problem Statement

Based on the Android market (AppBrain), there are many applications on Bluetooth function and usage. However, we have yet discovered application that allows the Smartphone users to recover back the lost Bluetooth connection. After reviewing some of the existing Bluetooth application (Bluetooth Finder (José 2011), Bluetooth Alarm (SSPARK 2011), Baby Bluetooth Alarm (Our Wireless World 2011), Bluetooth Tracker (Bowl 2011), Keep Bluetooth From Lost (Atcle 2011)) from (AppBrain), most of the applications couldn't fulfill all of the criteria below:

- **Scan Lost Device**

  Once a device gets out of range, the application will stop running and just prompt up a notification message letting users know that devices have been lost. However the application does not help user to search back on the lost device. So scanning is important to allow user to locate back the lost device whenever device gets back into range.

- **Alarm Triggering**

  This function is used to allow users to sound the alarm when users are close by. It will be difficult for user to find the lost device even if the lost device is in the same room. So by pressing a button, the lost device will make noise so that user will be able to locate back the lost device.

- **GeoLocation**

  GeoLocation concept is similar to Global Positioning System (GPS) (Chris 2009) and is use in navigating users from one place to another. In this function, longitude and latitude of the lost device will be sent back to the finder's device allowing the finder to search back the lost device by using the coordinates that he/she receives.

1.2 Objective

Therefore, the challenges that need to be overcome would include:-

- **To be able to auto-connect to lost pairing device**

  Once device got out of range, the application will continue on scanning for the lost device until the device is found. Once the device comes within range, the device will straight auto-connect to the lost device as long as the application and Bluetooth remains switch on.

- **To track back lost Bluetooth devices**

  Tracking back lost device can be solve by applying the alarm function. When user gets near to the lost device the Bluetooth connection will automatically connects back to the user's device. When this happen, user can track back the lost device just by a single press on a button in the application and this will trigger the alarm at the lost device side.

- **To locate lost Bluetooth connection**

  This can be solved by applying the GeoLocation functionality. It will help in searching for lost signal because once signal is lost it can either still in the same building or have been move out from the building. This will help to minimize down the search scope area for the users and other helpers as well and will results in faster recovery of the lost device.

1.3 Contribution

This application that I am proposing here would be able to help reduce the burden of the user of having the need to constantly checking on their belongings. The advantages would include helping to recover lost items more effectively. Lost devices can be detected and retrieve back as soon as possible. It will also help to reduce search area scope by using the GeoLocation functionality. Next this application will help to keep track of other devices so that once signal is out of range, signal recovery can start at once to search back lost devices. Lastly this application can also help to reduce search time by getting other users nearby to help search on the lost device. Once device have been found, notification will be send out to those helping in the search letting them know that the device have been found.

1.4 Background Information

After searching over the Android market, we found several applications that are quite similar to what we are planning to do. However, those applications are not suitable for the environment that we wish to implement. Applications such as Bluetooth Finder (José 2011), Bluetooth Alarm (SSPARK 2011), Baby Bluetooth Alarm (Our Wireless World 2011), Bluetooth Tracker (Bowl 2011) and Keep Bluetooth From Lost (Atcle 2011) that we found from the Android market all have slight similarities to what we are proposing to do here.

Bluetooth Finder uses the received signal strength indication (RSSI) to locate nearby Bluetooth devices. When device is near, the signal strength will be high and when the device is far away, the signal strength is low. For Bluetooth Alarm, the application will notify the user when the connected user goes out of range and it will also notify the user when the connected user is nearby. This application can be use together with headphones and PCs with Bluetooth technology. The next application that we found is the Baby Bluetooth Alarm. This application is made mainly for parents to keep track of their children. It can also be use to keep track of family members or friends in the public places. The application has two modes which is the tracker mode to keep track of the connected devices and the finder mode to search back the Bluetooth device. Besides that, we also found an application by the name of Bluetooth Tracker. This application is used to track connected devices whether they are within or out of range. The application has a function call the “Vibrate within Range” which is set by the user so that when the connected device get close by it will vibrate to notify the user. The other application that we found would be the Keep Bluetooth From Lost application. This application will ring whenever the connected device got disconnected for lost prevention.

From all these applications that we found, there are few similarities between them. All these similarities are a disadvantage from our point of view. One of them would be that neither of the applications included the GPS function. This function is needed in order to locate back the lost devices more effectively. The other similarities between them are that all these applications do not support tracking back of lost device. So when the other device got lost, the user will be prompted with a notification stating that device is lost. There are no actions taken to reconnect back the lost device.

2.0 Literature Review

Based on AppBrain, there are several applications that are quite similar to what we are proposing here. "Bluetooth Finder" (José 2011) is used to locate Bluetooth devices by applying the received signal strength indication (RSSI). When devices are close to each other the signal strength will be high but when the devices are away from each other over a far distance the signal strength will be low. Each devices found in the list will have their name, MAC id and a graphical signal strength meter with units in decibels (dBm) stated for each devices. The disadvantages of this application is that it will only show to the current user who is furthest away and who is the nearest. It will not prompt up notification when a signal is lost. This application also does not use the Global Positioning System (GPS) (Chris 2009) to locate the lost connection. So when another user goes out of range, it will be hard to find back the user. The other disadvantage is that this application does not support multiple object tracking. This application only will show to the user the other Bluetooth devices nearby, it does not make any connection with either of them.

The next application that is quite similar to our project would be the "Bluetooth Alarm" (SSPARK 2011). This Bluetooth Alarm will notify the user when another connected user is away from a certain range or when the other connected user is nearby. This application can be used for headphones and PCs with Bluetooth technology. It is design to be used in situation such as to keep track on their kids, reduce the chances of item lost such as bags and etc. For this application, the alarm will only ring at the main user (the one making the connection) when connection is far or near. The alarm does not ring on both sides of the end device. This application also does not use the Global Positioning System (GPS) (Chris 2009) to locate the lost connection. So it will be hard to track back the lost device when Bluetooth connection goes out of range. This application also does not support multiple object tracking. The device can only make a connection with another device one at a time.

"Baby Bluetooth Alarm" (Our Wireless World 2011) is also quite similar to our proposed application. The main usage of this application is to keep track of the children, family members or friends at a public area such as shopping mall, picnic or party. The application comes with two modes that is the “Tracker Mode” to keep track of the connected Bluetooth device so that if it goes out of range the alarm will ring and the “Finder Mode” to help search on the Bluetooth device so when the other device comes within range the alarm will ring as well. The disadvantage of this application is that only the main device will ring when the sub-devices go out or within range. The alarm does not ring on both end devices. This application also does not implement the Global Positioning System (GPS) (Chris 2009) to locate the lost connection. The application is used to let the user know when the sub-device comes within or goes out of range. Baby Bluetooth Alarm does not support multiple objects tracking as well, so only one connection can be made at a time.

“Bluetooth Tracker” (Bowl 2011) is also another application with some similar concept to our project. The application is used to determine whether the other connected device is within or out of range. For example, a device can be placed in the luggage, so when the luggage got far away from the user end device the phone will vibrate. Then when the luggage get near to you the phone will also vibrate if the user set the mode to “Vibrate Within Range”. The disadvantage of this application is that only the main device which is the phone mention above will vibrate, the sub-device place in the luggage will not vibrate or make any noise at all. So when the luggage comes within range user will have a hard time finding back the luggage. This application also does not implement the Global Positioning System (GPS) (Chris 2009) to locate the lost connection. So finding back the lost device will be quite difficult with this application. This application also does not support multiple objects tracking, so the application can only connect to a single end device at a time.

The other similar application would include “Keep Bluetooth From Lost” (Atcle 2011) as well. This application will sound the alarm when Bluetooth device got disconnected for lost prevention. For example, the device will ring the alarm when the Bluetooth headset got disconnected. So this application will also ring the alarm when the other Bluetooth device got disconnected. The application does not use the Global Positioning System (GPS) (Chris 2009) to locate the lost connection, so locating the last device will be quite difficult. This application does not support multiple objects tracking, so only can make a connection with an end device at a single time.

From the application that we have gathered on, we can see that most of the application has some similarities with what we are proposing here. However, most of the application does not support Global Positioning System (GPS) (Chris 2009) and multiple objects tracking which can be useful when searching back on lost devices and enable the application to make more than one connection with different Bluetooth devices at a time.

Function \ Software	Bluetooth Finder	Bluetooth Alarm	Baby Bluetooth Alarm	Bluetooth Tracker	Keep Bluetooth From Lost
Auto-reconnection	X	X	X	√	X
Notification (Message, Vibration, Alarm)	X	√	√	√	√
GeoLocation	X	X	X	X	X
Multiple Object Tracking	X	X	X	X	X

Table 2.0.1 : Software vs Functionality

Table 2.0.1 shows the similar software that we found from the Android market and we use these applications to compare them with several functions that we want to implement into our project. After reviewing the application with the functions that they can accomplish, we found that most of the application does not implement the auto-reconnection function. So whenever a Bluetooth device goes out of range, the application will only prompt to let the user know that device got out of range. The application will never try to reconnect back the lost device even when the device comes back within range.

The next function that we are comparing is the notification functions. Most of the application uses the notification functions to interact with the user except for the Bluetooth finder which need the user to see the screen for any changes. It is good that most application uses the notification function as this is a good way to interact and to get the users attention whenever someone/something goes out of range.

As for the GeoLocation function, none of the applications apply it in their application. Even those application that need to search back on lost connection such as the Baby Bluetooth Alarm and the Bluetooth Tracker which provide function for the user to search back the lost connection did not apply in GeoLocation functionality. They use the proximity detection to search on the Bluetooth connection so whenever the Bluetooth connections come back into the range it will notify the user. This is not an appropriate way of locating back lost belonging as the search area is too wide.

The last function that we compare is the multiple objects tracking functionality. Multiple object tracking is use to connect multiple devices at the same time so that to allow multiple object tracking and searching. Most of the application that we find uses one-to-one connection so only one connection with the end device can be made at a particular time.

2.1 Application Functions

The software application that we propose to build is a Bluetooth Recovery application. The main objective for this application design is to enable users to recover back a lost Bluetooth connection. Some research has been done on the available application that is similar to what we are proposing here. However, there are only applications that notify the users whenever the Bluetooth connection is gone. Therefore, we would like to add in some additional functionalities to the existing Bluetooth application that would enable users to recover back or find back the lost Bluetooth connection which we will explain more in detail below:-

1) Auto-reconnection

   This additional function will allows user to automatically reconnect to the lost device whenever the lost device gets within the bluetooth range. This function is important as it allows user to be notified when the lost device is nearby. It will also prompt up notification to the user when auto-connection is successful.

2) GeoLocation

   In order to find back the lost Bluetooth connection, this GeoLocation will come in handy. This function is used to send back the location which is the longitude and the latitude of the lost device to the user. This can be done by using the messaging function. For example, when a device got lost user can send a message to the lost device which will immediately trigger the GeoLocation application and the application will automatically send back the location of the lost device back to the user.

3) Alarm Trigger

   After GeoLocation had located the lost Bluetooth connection, user will then use this function to continue on searching for the lost connection. After the application found the exact location of the lost connection, user can use this function to narrow down their search scope by pressing on the alarm button continuously so that the lost device can continue on making noise allowing the user to locate the lost device easier and more effectively.

3.0 Methodology

We will be implementing the spiral model (David 2012) to build our application. Refer to Figure 3.0.1 below.

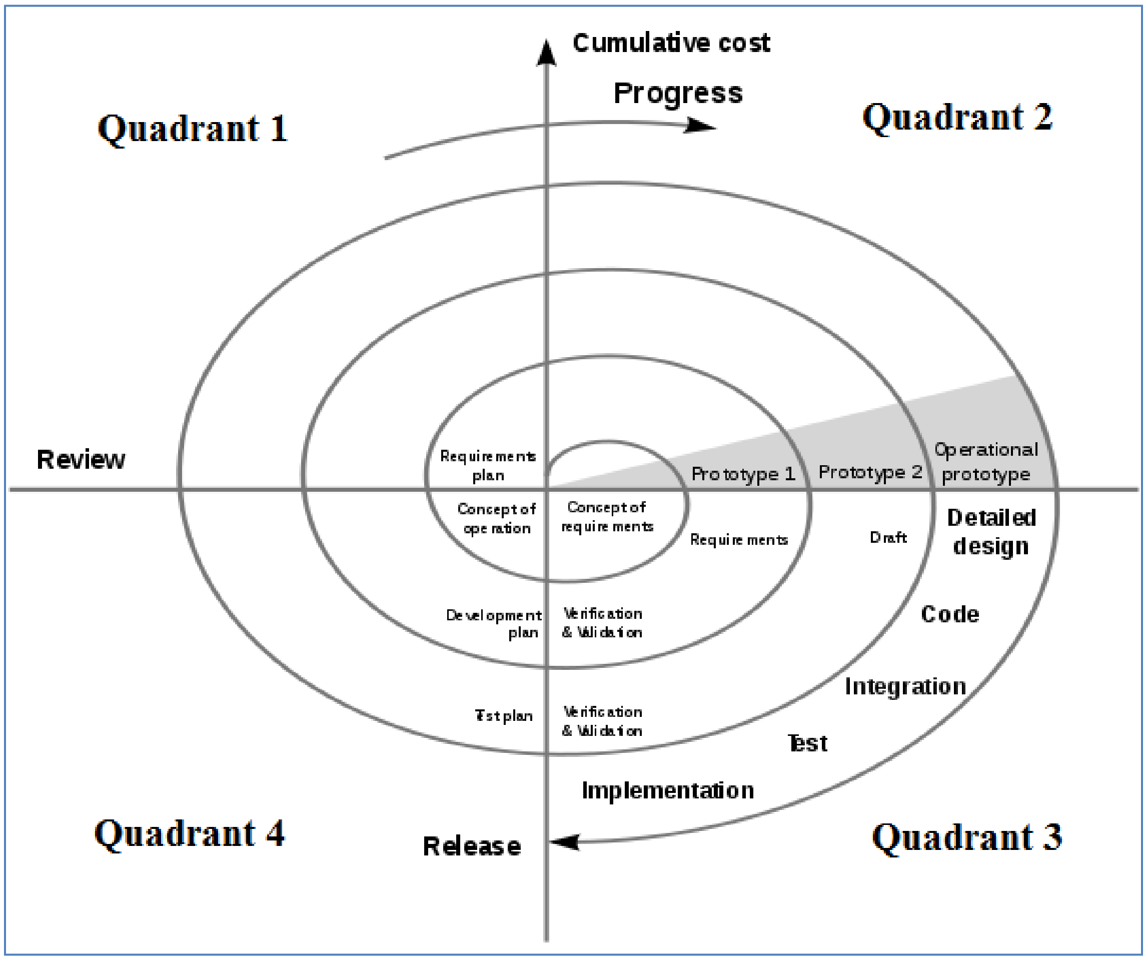


Figure 3.0.1 : Spiral Model

Quadrant 1

- Gather information of current existing systems.
- Find out more information on current Bluetooth and the range of the Bluetooth device.

Quadrant 2

- Design the flow of the application. How it is going to function.
- Design the interface of the application.

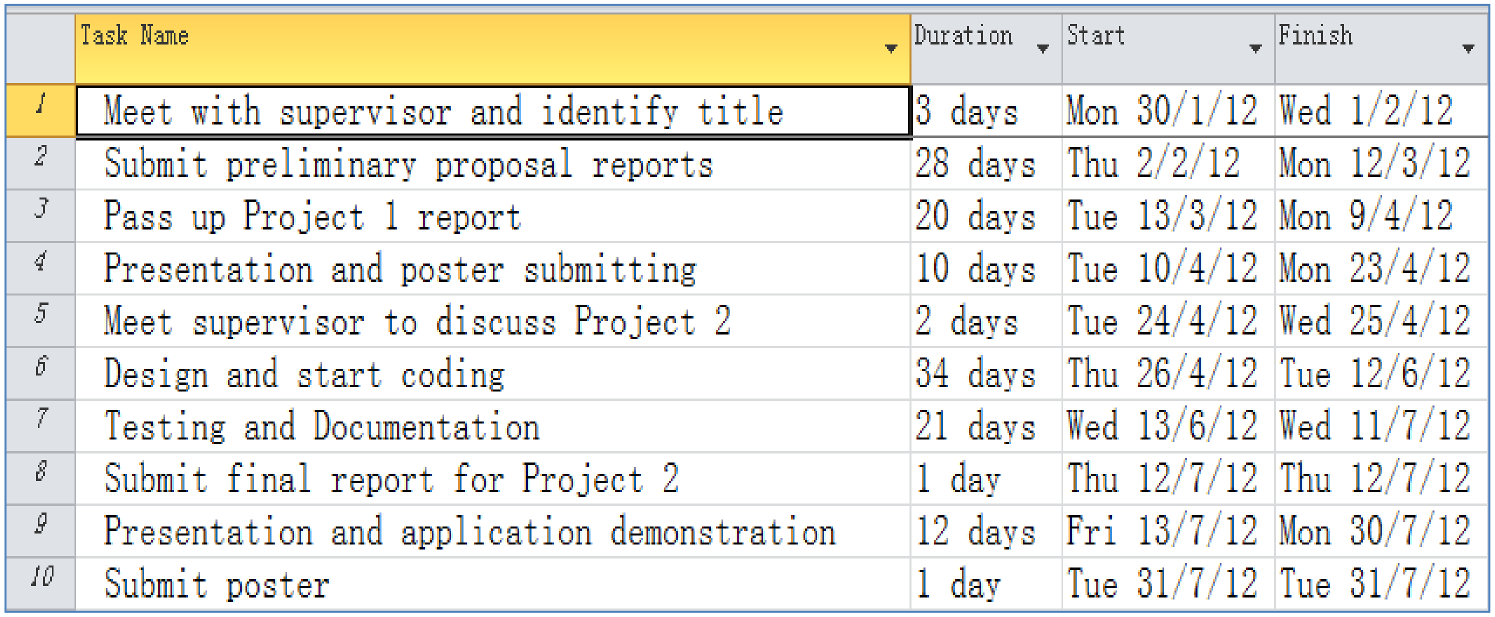
Quadrant 3

- Code a simple Bluetooth application.
- Further enhance the application with more functions. (example: Alarm, message, vibration)

Quadrant 4

- Combine everything up and test runs it to ensure no bug and error.
- Test the application on different types of mobile phones. Find out the OS needed to get all the functions working.
- Try to further improve the system if possible.
- Documentation.

3.1 Timeline



	Task Name	Duration	Start	Finish
1	Meet with supervisor and identify title	3 days	Mon 30/1/12	Wed 1/2/12
2	Submit preliminary proposal reports	28 days	Thu 2/2/12	Mon 12/3/12
3	Pass up Project 1 report	20 days	Tue 13/3/12	Mon 9/4/12
4	Presentation and poster submitting	10 days	Tue 10/4/12	Mon 23/4/12
5	Meet supervisor to discuss Project 2	2 days	Tue 24/4/12	Wed 25/4/12
6	Design and start coding	34 days	Thu 26/4/12	Tue 12/6/12
7	Testing and Documentation	21 days	Wed 13/6/12	Wed 11/7/12
8	Submit final report for Project 2	1 day	Thu 12/7/12	Thu 12/7/12
9	Presentation and application demonstration	12 days	Fri 13/7/12	Mon 30/7/12
10	Submit poster	1 day	Tue 31/7/12	Tue 31/7/12

Figure 3.1.1 Gantt Chart

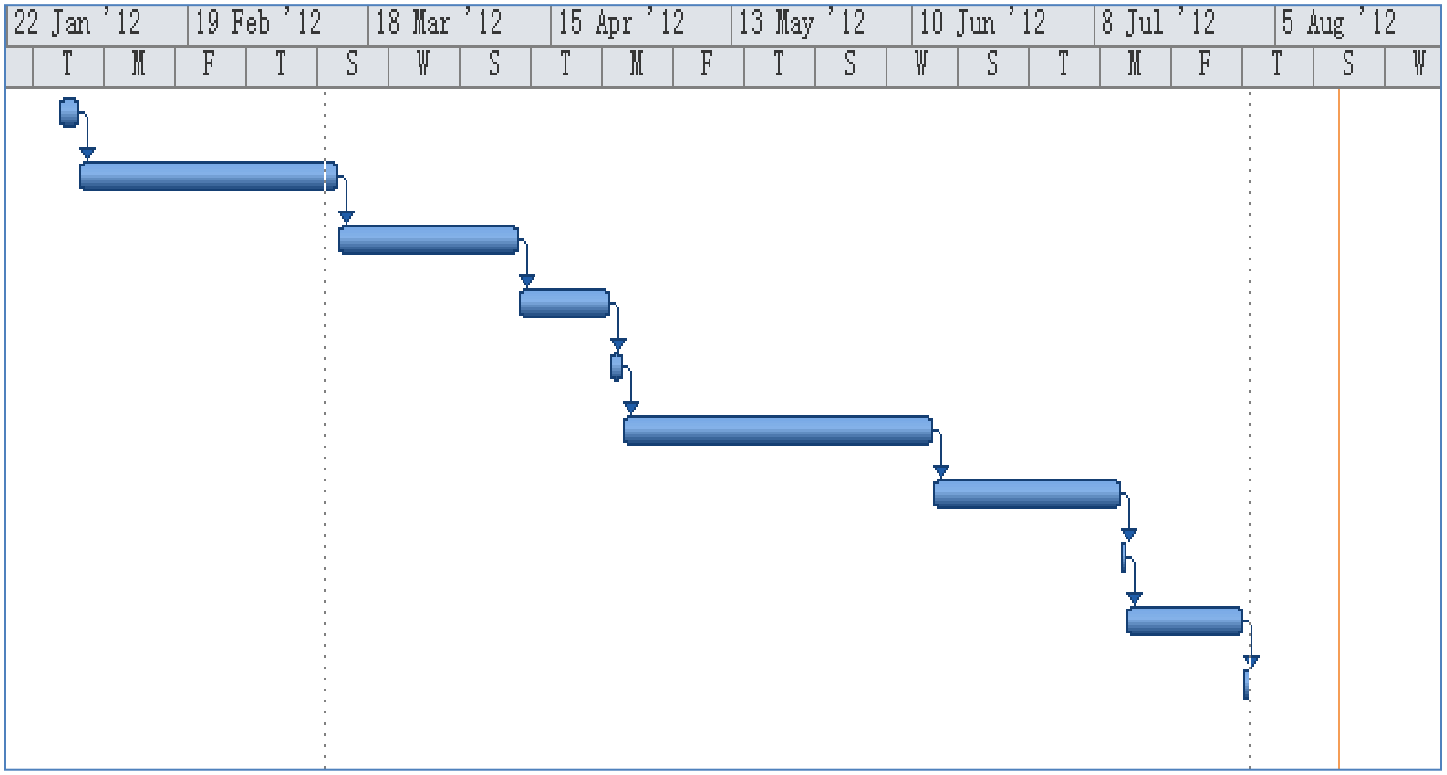


Figure 3.1.2 Gantt Chart

3.2 Technology Involved

- **Bluetooth**

  Technology involved will include the Bluetooth technology (Bluetooth SIG). This is the main technology used for our application. Bluetooth had been widely implemented in a lot of different devices and that includes mobile devices. Most of the mobile devices nowadays have been included with Bluetooth technology to enable faster sharing of information between two devices.

- **Global Positioning System**

  The other technology is the Global Positioning System (GPS) (Chris 2009) which have been widely used in navigation. By using this technology, it will allow the lost devices to be located in situation where the devices go out of range.

- **Android OS**

  Android is a Linux based operating system for mobile devices. We choose Android because Android is an open source software and Android powered devices are part of the best-selling smartphones in the market nowadays. So this application will be useful to quite a number of users.

3.3 Requirement Specification

3.3.1 Class Diagram

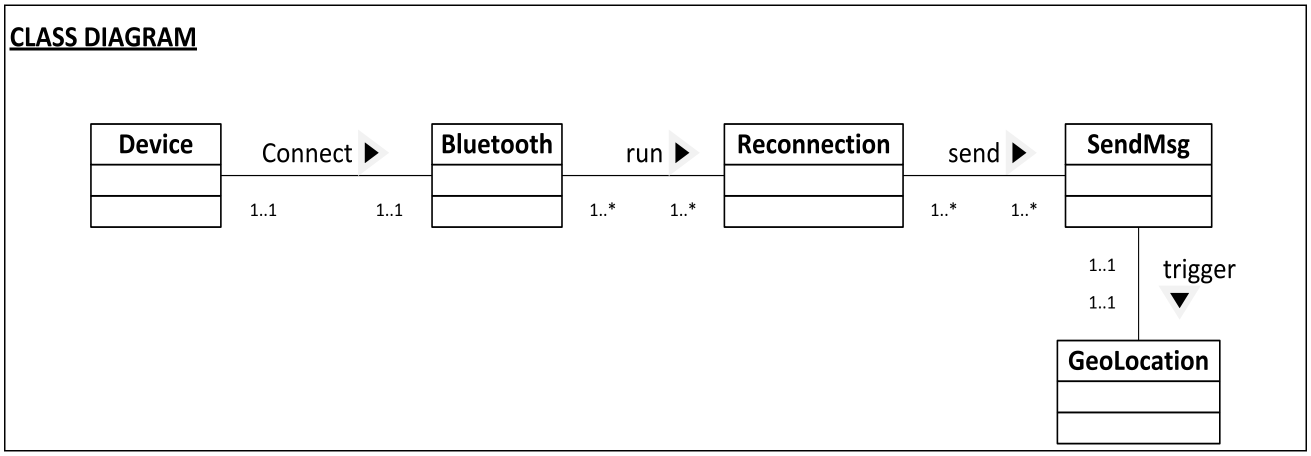


Figure 3.3.1 : Class Diagram

The class diagram is used to describe the data type and the relationships between each class. There are 5 classes which is Device, Bluetooth, Reconnection, SendMsg and GeoLocation class. One device can connect to one Bluetooth port in the mobile phone itself and one Bluetooth port can only be added to one device at a time. One and many Bluetooth can run many reconnection function and reconnection function can be used by one and many Bluetooth port. Reconnection function will trigger one or many SendMsg function to allow message notification to be sent out to either one or many users. SendMsg function can only send to one GeoLocation function at one time and GeoLocation function can also only be triggered by one SendMsg function at one time.

3.3.2 Entity-Relationship Diagram

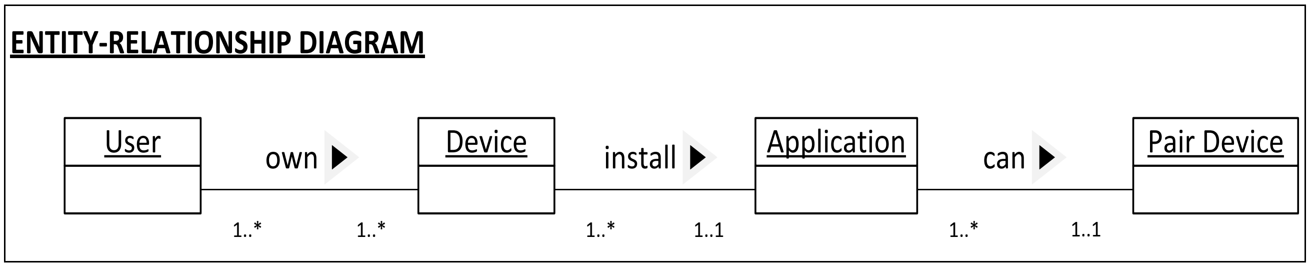


Figure 3.3.2 : Entity-Relationship Diagram

Entity-relationship diagram (ER diagram) is used to shows the logical structure of the databases. One or many users can own many devices and one or many devices can be owned by one or many users. One device can only install one type of application at a time and the same application can only be install into one device. The application can connect to one or many pair devices at the same time and one or many pair devices can also connect to the application.

3.3.3 Use Case Diagram

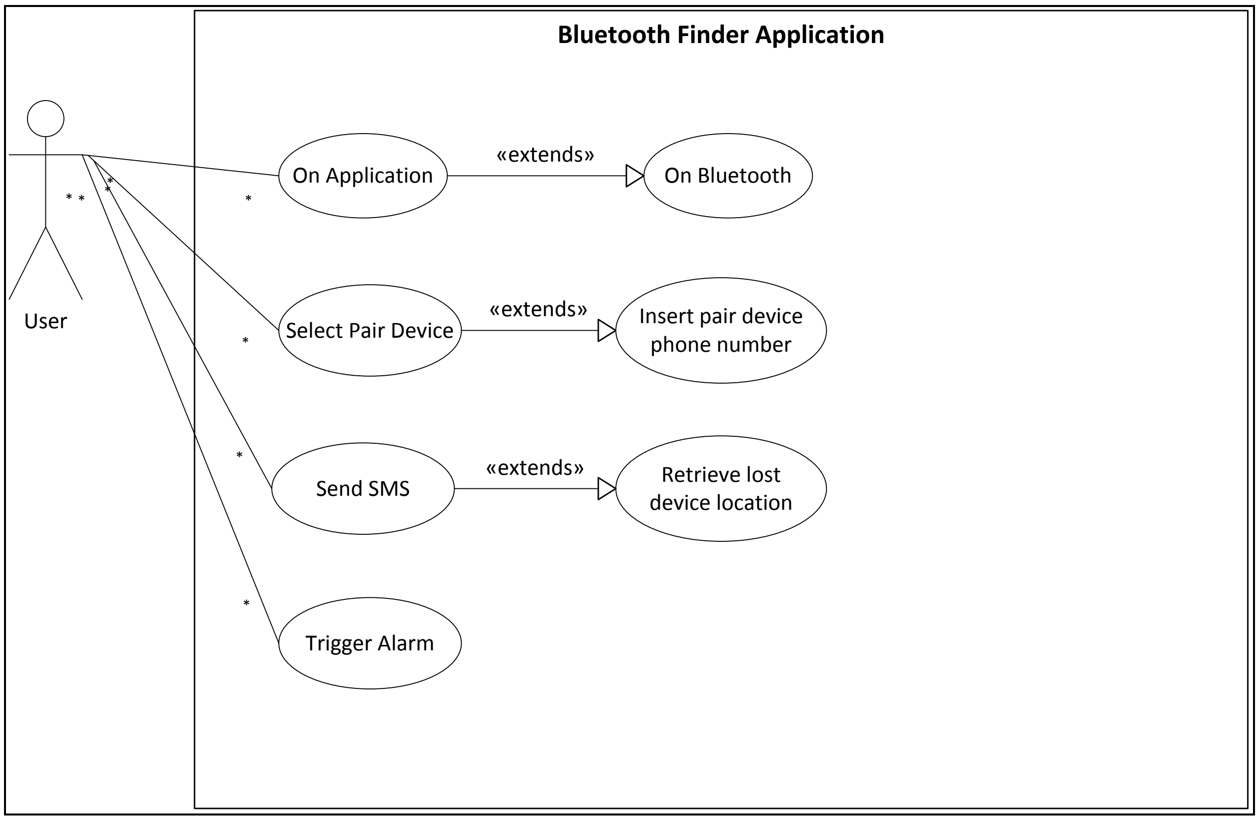


Figure 3.3.3 : Use case model

Use case diagram is use to show how the user interacts with the system. This system is called the Bluetooth recovery application. User on the application and then connect to the Bluetooth in order for the application to work. Then user will need to pair with other devices by selecting which device they wish to pair their mobile device with. Once connection is lost, the user can trigger the send SMS function. In this function, user can send a message with the key word to the lost device which will then automatically send back the location link of the lost device back to the user. Once user found the location of the lost device and user is nearby, user can trigger the alarm of the lost device by pressing on an alarm button so that the lost device will make noise to allow user to know where the lost device exact location is.

3.3.4 Sequence Diagram

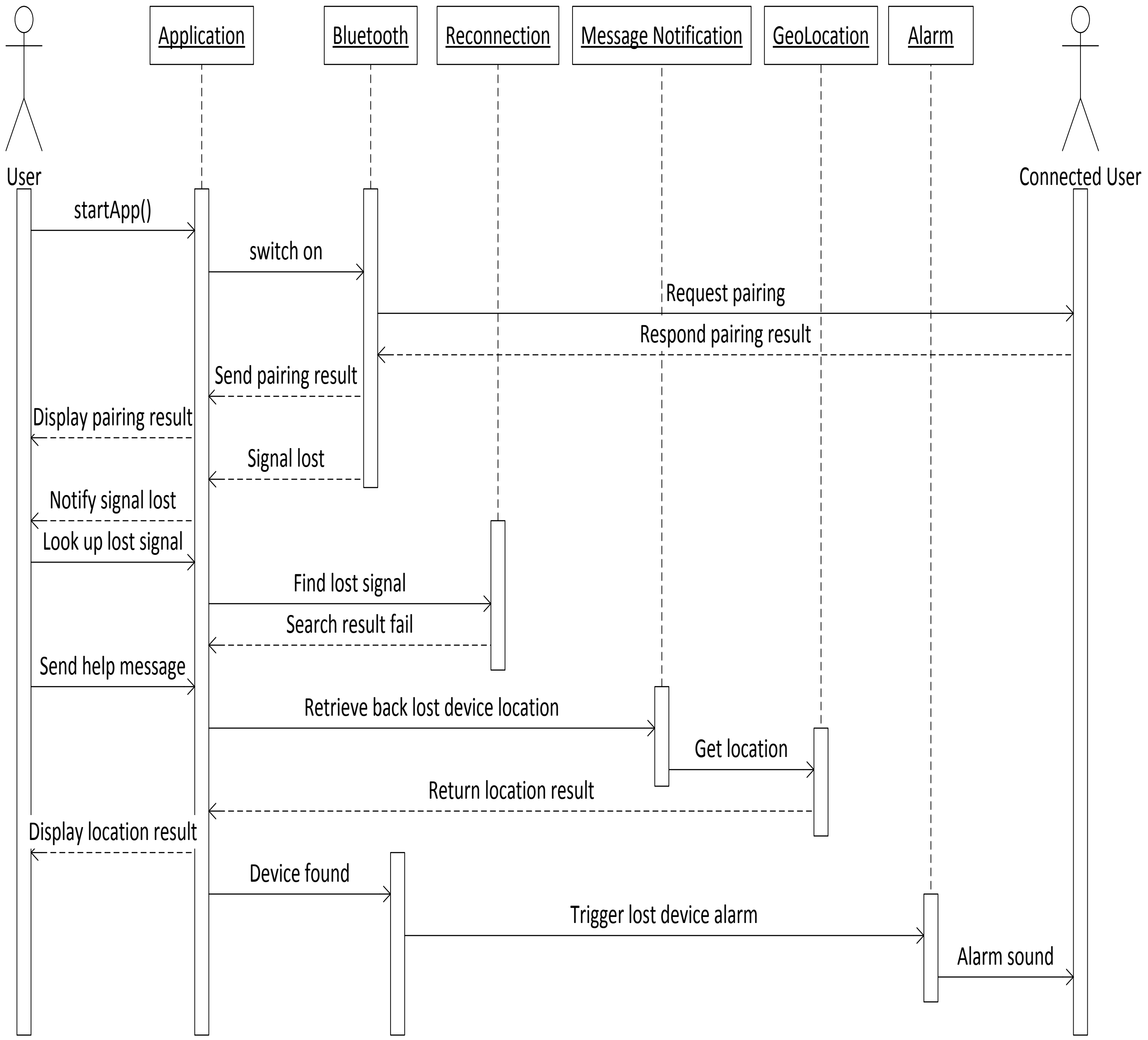


Figure 3.3.4 : Sequence Diagram

The sequence diagram model the collaboration of objects based on a time sequence. User uses the application by first starting up the application. Then user will need to switch on the Bluetooth. Bluetooth enable pairing with another user device by requesting pairing with another user. The connected user will then responds and send back the pairing result back to the Bluetooth function. Bluetooth will send back the pairing result to the application and the application will then display the pairing result to the user. Once the Bluetooth signal goes out of range, the Bluetooth function will send a signal lost message to the application so that the application can notify the user that pairing connection have been lost. The application itself will continue on trying to reconnect to the lost device. If reconnection is unsuccessful, user can send out help message to the lost device to retrieve back the location of the lost device via the GeoLocation functionality. This function will automatically send back the location of the lost device once it receive the correct key word from the sender. When user found the device and user is nearby to the lost device, user can trigger the alarm by pressing on the alarm button to get the lost device to make some noise so user can know where exactly the device is at.

3.3.5 Activity Diagram

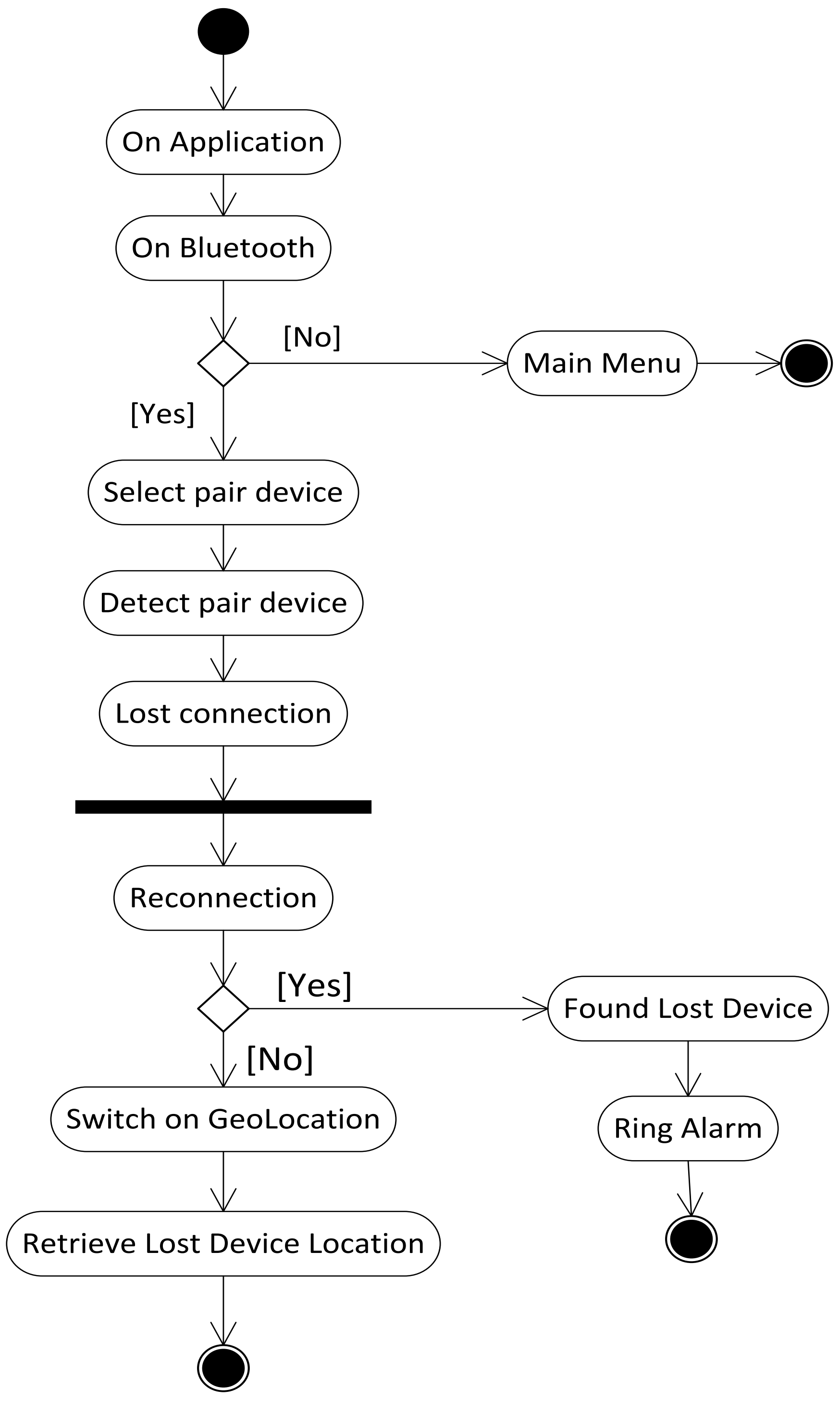


Figure 3.3.5 : Activity Diagram

Activity diagram shows the work flow from the starting point to the finish point. From the start point, user will on the application. Then, user will need to turn on the Bluetooth. If user refuse to turn on the Bluetooth, then the application will back to the main menu and end unless user turn on the Bluetooth. Next, user will need to detect and select the pairing device. Once connection is lost then user can try to walk around to try to get the lost device to reconnect back to the user's device. If still couldn't find then user can send out a message to get the Geolocation function working which will automatically send back the location of the lost device. When user found the lost device, user can then press a button to ring the lost device alarm so user can know where the device exact location is.

3.4 Test Plan

We will be implementing both black-box (Software Testing Help) (Laurie 2006) and white-box testing (Build Security In) (Laurie 2006) (Software Testing Help). Black-box testing is use to test out the functionality of the application. We will be using black-box testing to test out the application function such as detecting Bluetooth range, Bluetooth reconnection function, GeoLocation function, message notification function, sending help message to other user, and etc.

On the other hand, white-box testing is a method to test out the internal structures of an application. It will be used to test out our project to ensure that the program is coded correctly. The program will be tested out by testing its independent path to ensure that the path towards each function and back are in the correct order. Then logical decisions will be tested to make sure that the logic of the program is properly written. White-box testing is implemented to filter out the bugs and errors within the program itself. This is because when we are writing our program, we would not know whether the logic behind what we have in our mind and the one written inside the program will work the same. We will use white-box testing to test out our application by inserting different types of data and see whether the output is the expected output and not the wrong ones. If the output is wrong it means that something is wrong with the programming logic and needs to be corrected.

4.0 Implementation

In this project, we will be building this application by using Eclipse Classic 4.2. This is because eclipse is an open source tool allowing users to build their own application free of charge. Eclipse can automatically build our application and at the same time allows us to debug your own project with a debug key. Eclipse also allows us to test out our own software by installing our application into our own android device using the USB port. We will also be using platform 2.3.6 which is also known as GingerBread to run our application because this platform is supported by most Android devices in the market. As for the Bluetooth version, we will be based on version 2.0 which is the version of our own Android device which we are using to test out this project.

Figure 4.0.1 below illustrates how the application functions. As we can see from the diagram, reconnection will start once the pairing device Bluetooth signal is lost or had been interrupted. Besides that, user can locate back the lost device location by sending over a message to the lost device to ask where the current location of the lost device is. User can also press on the alarm button to trigger the alarm at the lost device side. All this will be explained further on at the coding part.

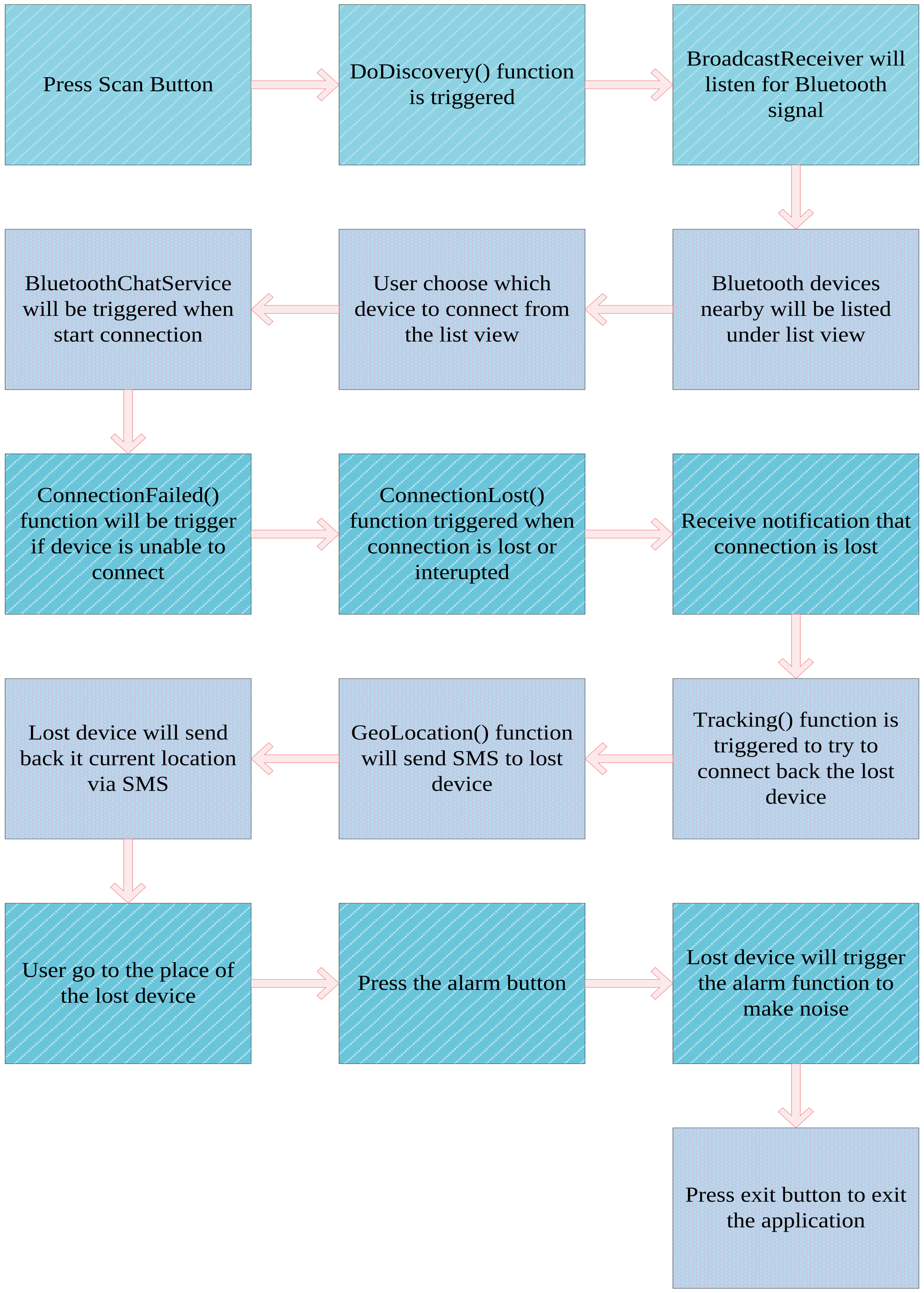
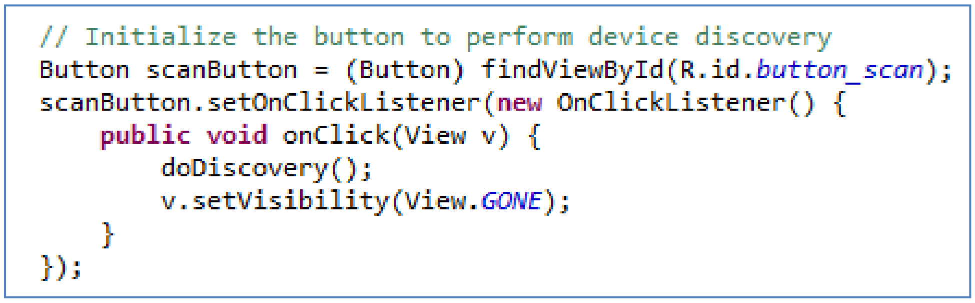

Press Scan Button
DoDiscovery() function is triggered
BroadcastReceiver will listen for Bluetooth signal
Bluetooth devices nearby will be listed under list view
User choose which device to connect from the list view
BluetoothChatService will be triggered when start connection
ConnectionFailed() function will be trigger if device is unable to connect
ConnectionLost() function triggered when connection is lost or interupted
Receive notification that connection is lost
Tracking() function is triggered to try to connect back the lost device
GeoLocation() function will send SMS to lost device
Lost device will send back it current location via SMS
User go to the place of the lost device
Press the alarm button
Lost device will trigger the alarm function to make noise
Press exit button to exit the application

Figure 4.0.1 : The Flow of the Application

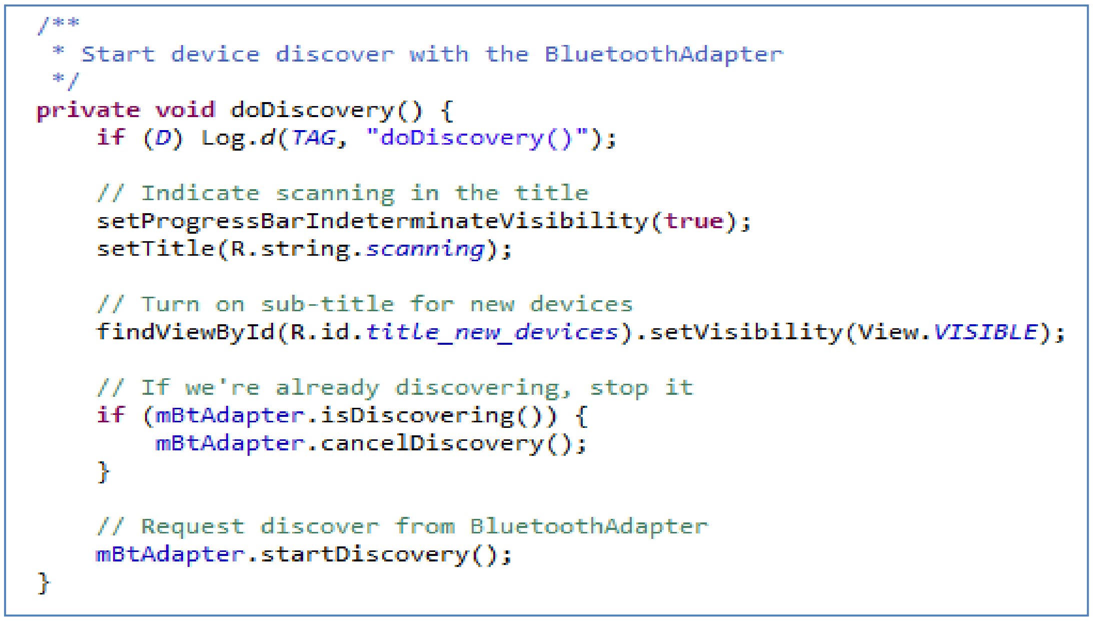
4.1 Implementation of the Bluetooth Discovery Process



```java
// Initialize the button to perform device discovery
Button scanButton = (Button) findViewById(R.id.button_scan);
scanButton.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        doDiscovery();
        v.setVisibility(View.GONE);
    }
});
```

Figure 4.1.1 : Source code for Scan Button

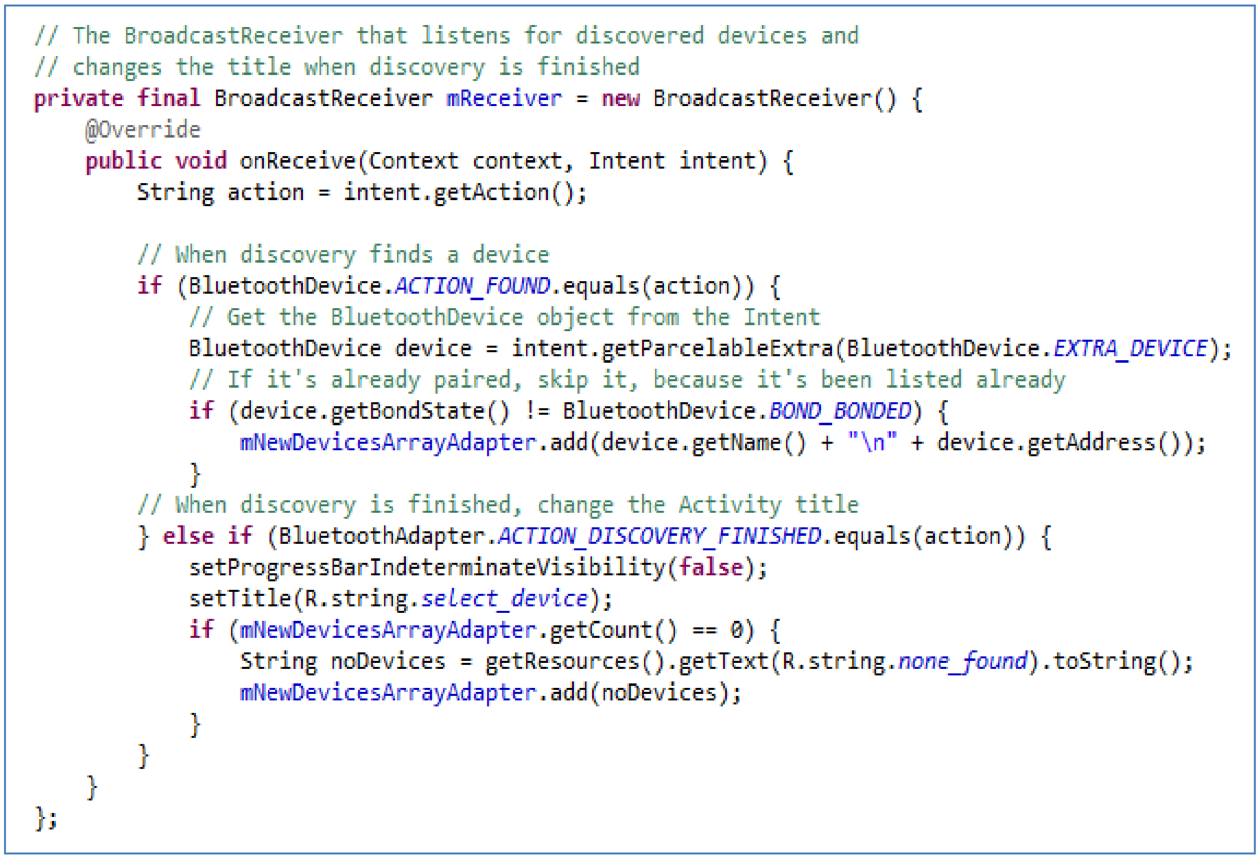
Figure 4.1.1 shows the coding for a scanning button. When this scanButton is pressed by user, it will trigger the class doDiscovery() below to scan for nearby devices with Bluetooth function switch on.



```java
/**
 * Start device discover with the BluetoothAdapter
 */
private void doDiscovery() {
    if (D) Log.d(TAG, "doDiscovery()");

    // Indicate scanning in the title
    setProgressBarIndeterminateVisibility(true);
    setTitle(R.string.scanning);

    // Turn on sub-title for new devices
    findViewById(R.id.title_new_devices).setVisibility(View.VISIBLE);

    // If we're already discovering, stop it
    if (mBtAdapter.isDiscovering()) {
        mBtAdapter.cancelDiscovery();
    }

    // Request discover from BluetoothAdapter
    mBtAdapter.startDiscovery();
}
```

Figure 4.1.2 : Source code for doDiscovery

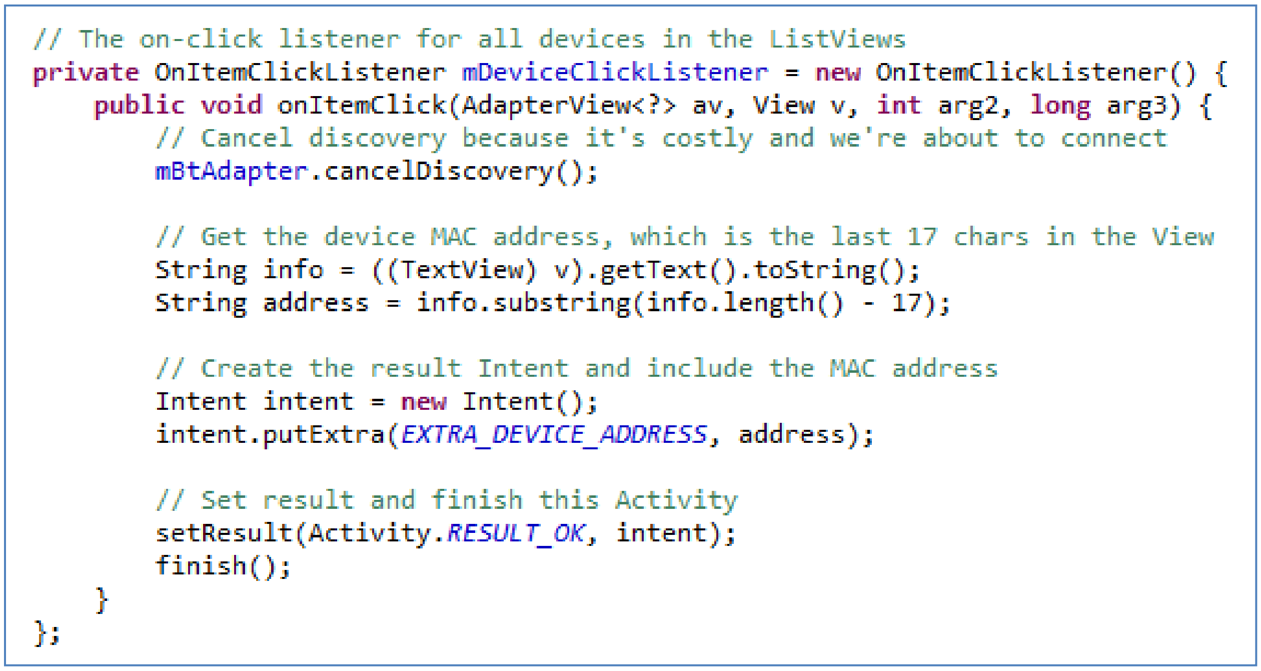
This class is initiated by the “Scan for device” button. When user pressed on this button it will start to discover all nearby devices with the Bluetooth adapter. After scanning end, all the nearby devices with Bluetooth switch on will be listed down under “Other Available Devices” list view.



```java
// The BroadcastReceiver that listens for discovered devices and
// changes the title when discovery is finished
private final BroadcastReceiver mReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();

        // When discovery finds a device
        if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            // Get the BluetoothDevice object from the Intent
            BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            // If it's already paired, skip it, because it's been listed already
            if (device.getBondState() != BluetoothDevice.BOND_BONDED) {
                mNewDevicesArrayAdapter.add(device.getName() + "\n" + device.getAddress());
            }
        // When discovery is finished, change the Activity title
        } else if (BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(action)) {
            setProgressBarIndeterminateVisibility(false);
            setTitle(R.string.select_device);
            if (mNewDevicesArrayAdapter.getCount() == 0) {
                String noDevices = getResources().getText(R.string.none_found).toString();
                mNewDevicesArrayAdapter.add(noDevices);
            }
        }
    }
};
```

Figure 4.1.3 : Source code for BroadcastReceiver

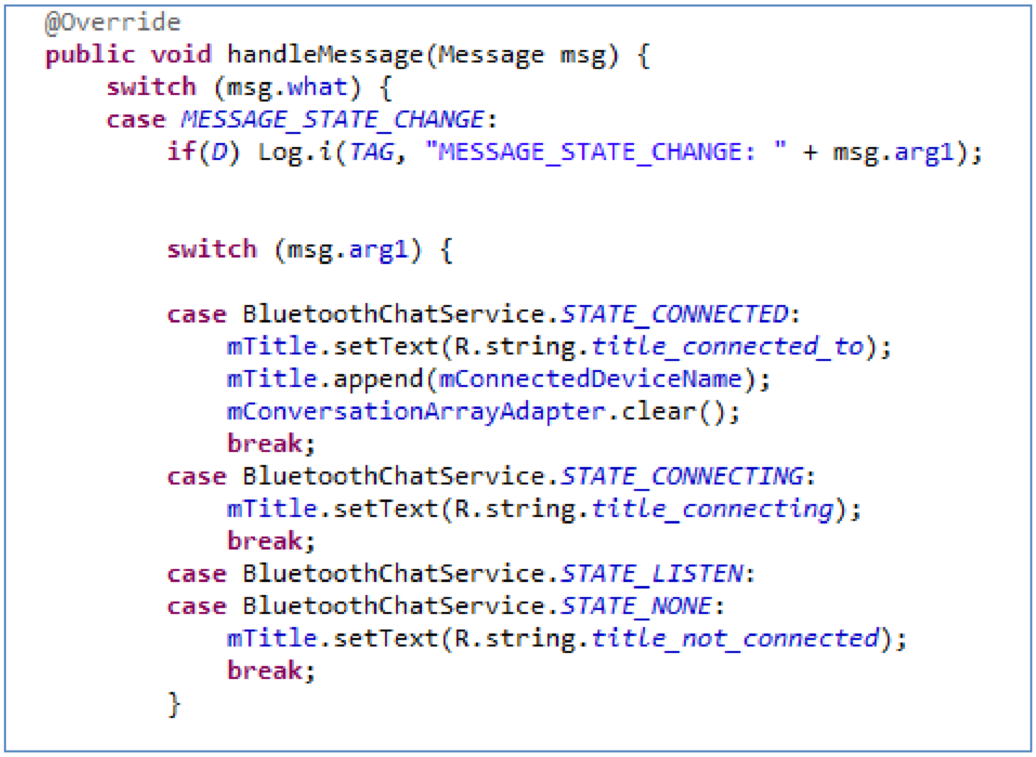
BroadcastReceiver class is used to listen for Bluetooth connection. When BroadcastReceiver found a device, it will list down the device name and the device MAC addresses into the list view so that user knows what devices are available around them as long as the device Bluetooth is switch on. If no devices are found, then no devices will be listed out.



```
// The on-click listener for all devices in the ListViews
private OnItemClickListener mDeviceClickListener = new OnItemClickListener() {
    public void onItemClick(AdapterView<?> av, View v, int arg2, long arg3) {
        // Cancel discovery because it's costly and we're about to connect
        mBtAdapter.cancelDiscovery();

        // Get the device MAC address, which is the last 17 chars in the View
        String info = ((TextView) v).getText().toString();
        String address = info.substring(info.length() - 17);

        // Create the result Intent and include the MAC address
        Intent intent = new Intent();
        intent.putExtra(EXTRA_DEVICE_ADDRESS, address);

        // Set result and finish this Activity
        setResult(Activity.RESULT_OK, intent);
        finish();
    }
};
```

Figure 4.1.4 : Source code for list view listener

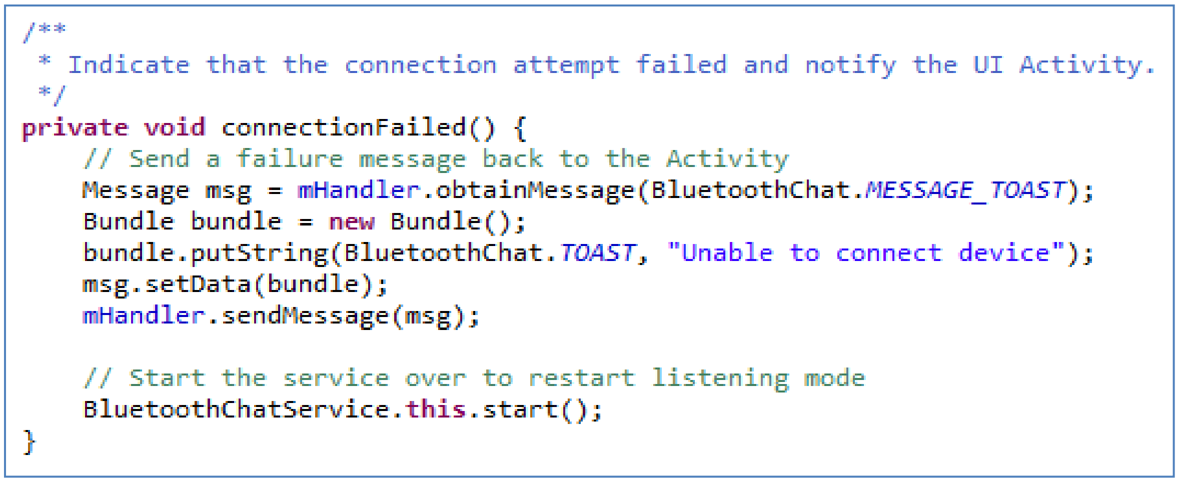
This listener is initiated when user pressed on the device that user wish to connect to on either the “Paired Devices” or the “Other Available Devices” list view. Once pressed, the listener will get the device name together with the device MAC address and store it into an intent which will later be passed on to another activity to be process.



```
@Override
public void handleMessage(Message msg) {
    switch (msg.what) {
    case MESSAGE_STATE_CHANGE:
        if(D) Log.i(TAG, "MESSAGE_STATE_CHANGE: " + msg.arg1);


        switch (msg.arg1) {

        case BluetoothChatService.STATE_CONNECTED:
            mTitle.setText(R.string.title_connected_to);
            mTitle.append(mConnectedDeviceName);
            mConversationArrayAdapter.clear();
            break;
        case BluetoothChatService.STATE_CONNECTING:
            mTitle.setText(R.string.title_connecting);
            break;
        case BluetoothChatService.STATE_LISTEN:
        case BluetoothChatService.STATE_NONE:
            mTitle.setText(R.string.title_not_connected);
            break;
        }
```

Figure 4.1.5 : Source code for Bluetooth connection state

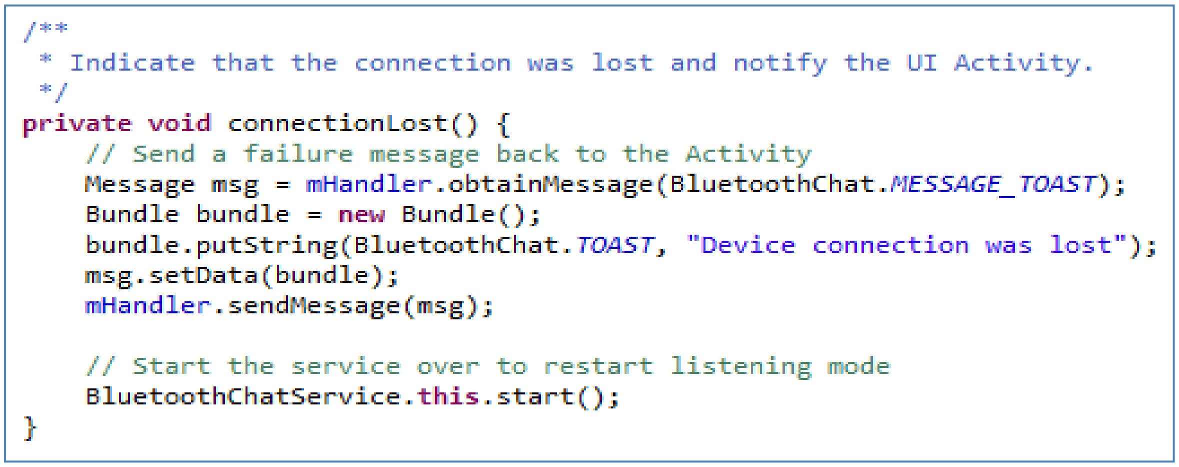
This is the message handler that will allow user to know what is happening to the connection. There is a small title bar at the top of our user interface that will show out to the user whenever the state of the connection changes.



```
/**
 * Indicate that the connection attempt failed and notify the UI Activity.
 */
private void connectionFailed() {
    // Send a failure message back to the Activity
    Message msg = mHandler.obtainMessage(BluetoothChat.MESSAGE_TOAST);
    Bundle bundle = new Bundle();
    bundle.putString(BluetoothChat.TOAST, "Unable to connect device");
    msg.setData(bundle);
    mHandler.sendMessage(msg);

    // Start the service over to restart listening mode
    BluetoothChatService.this.start();
}
```

Figure 4.1.6 : Source code for connection fail

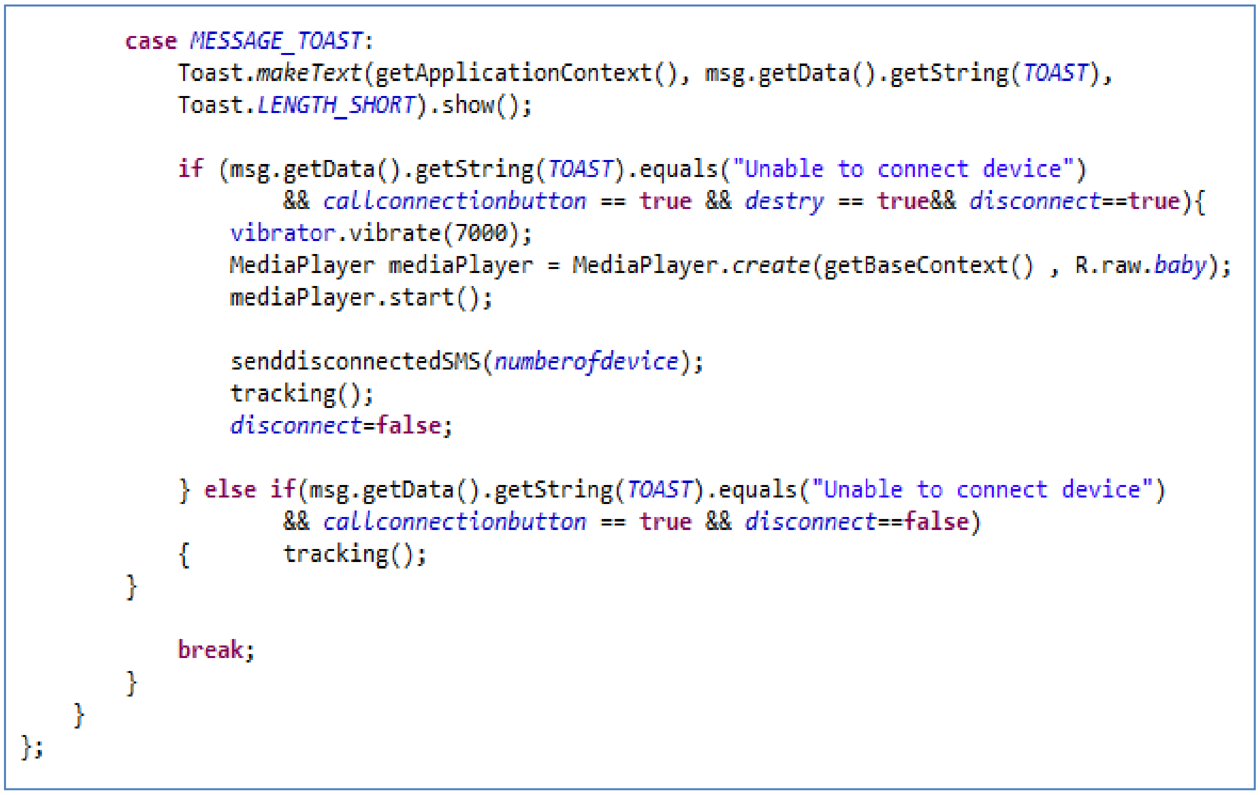
This connectionFailed() class is used to notify the user whenever the Bluetooth connection fail to connect. It will returns back a string "Unable to connect device" to the user interface so that user knows that the device are unable to connect to each other.



```
/**
 * Indicate that the connection was lost and notify the UI Activity.
 */
private void connectionLost() {
    // Send a failure message back to the Activity
    Message msg = mHandler.obtainMessage(BluetoothChat.MESSAGE_TOAST);
    Bundle bundle = new Bundle();
    bundle.putString(BluetoothChat.TOAST, "Device connection was lost");
    msg.setData(bundle);
    mHandler.sendMessage(msg);

    // Start the service over to restart listening mode
    BluetoothChatService.this.start();
}
```

Figure 4.1.7 : Source code for connection lost

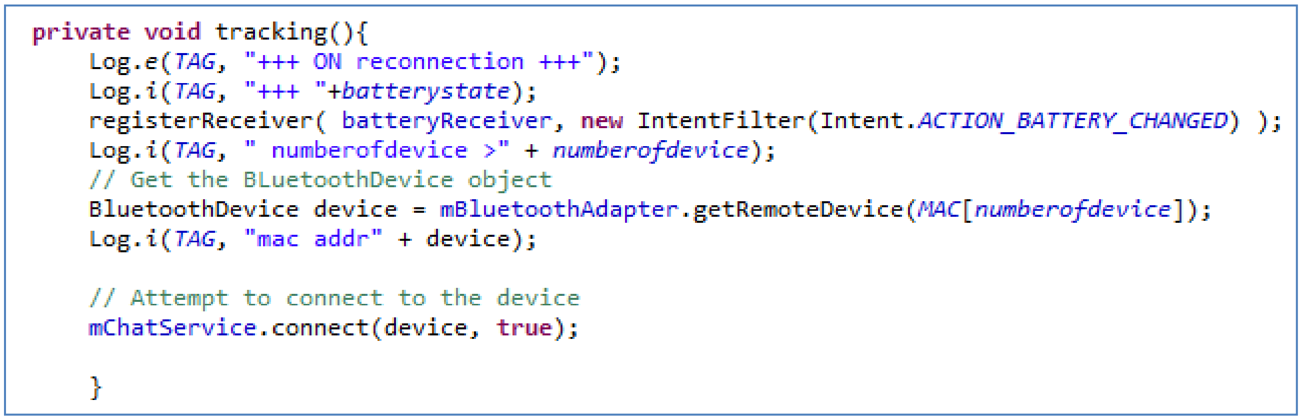
For this connectionLost() class, it will be triggered when connection between two or more devices got lost. This happens when Bluetooth connection went out of range or when Bluetooth connection is turn off when pairing is on-going. So this class will returns a string "Device connection was lost" to the user interface to let the user knows that device connection had been interrupted.



```
        case MESSAGE_TOAST:
            Toast.makeText(getApplicationContext(), msg.getData().getString(TOAST),
            Toast.LENGTH_SHORT).show();

            if (msg.getData().getString(TOAST).equals("Unable to connect device")
                    && callconnectionbutton == true && destry == true&& disconnect==true){
                vibrator.vibrate(7000);
                MediaPlayer mediaPlayer = MediaPlayer.create(getBaseContext() , R.raw.baby);
                mediaPlayer.start();

                senddisconnectedSMS(numberofdevice);
                tracking();
                disconnect=false;

            } else if(msg.getData().getString(TOAST).equals("Unable to connect device")
                    && callconnectionbutton == true && disconnect==false)
            {       tracking();
        }

            break;
        }
    }
};
```

Figure 4.1.8 : Source code for disconnection

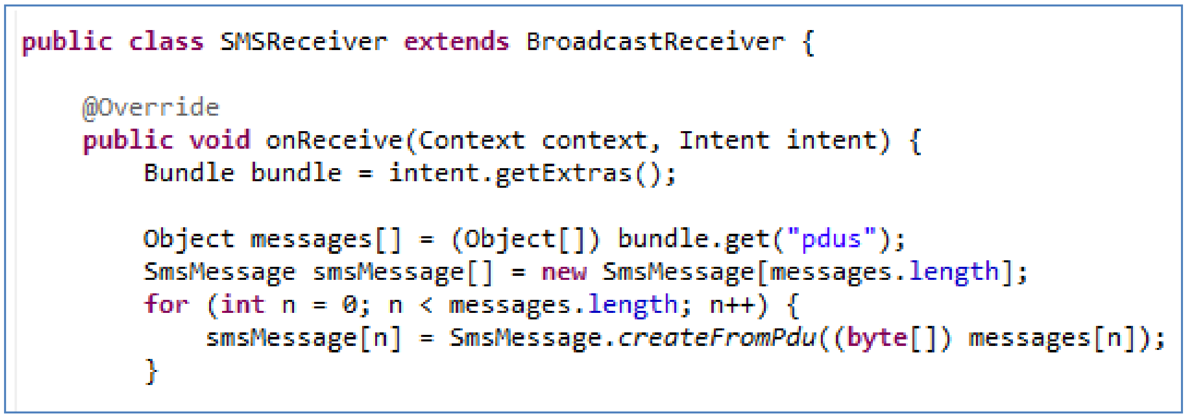
This is where reconnection happens. When the message "Unable to connect device" is passed into here, the device will sound the alarm to alert the user that connection had been interrupted. Then it will continue to search back the lost device signal and try to reconnect back if the device is available.



```java
private void tracking(){
    Log.e(TAG, "+++ ON reconnection +++");
    Log.i(TAG, "+++ "+batterystate);
    registerReceiver( batteryReceiver, new IntentFilter(Intent.ACTION_BATTERY_CHANGED) );
    Log.i(TAG, " numberofdevice >" + numberofdevice);
    // Get the BLuetoothDevice object
    BluetoothDevice device = mBluetoothAdapter.getRemoteDevice(MAC[numberofdevice]);
    Log.i(TAG, "mac addr" + device);

    // Attempt to connect to the device
    mChatService.connect(device, true);

    }
```

Figure 4.1.9 : Source code for reconnection

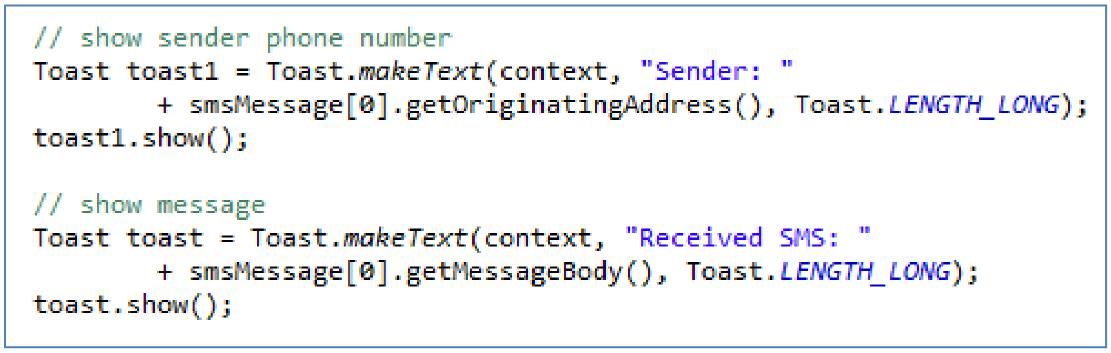
This tracking() class is use to reconnect back to the lost device. When the lost device MAC address is found, the mChatService will trigger the BluetoothChatService to automatically connect back to the lost device.



```java
public class SMSReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        Bundle bundle = intent.getExtras();

        Object messages[] = (Object[]) bundle.get("pdus");
        SmsMessage smsMessage[] = new SmsMessage[messages.length];
        for (int n = 0; n < messages.length; n++) {
            smsMessage[n] = SmsMessage.createFromPdu((byte[]) messages[n]);
        }
```

Figure 4.1.10 : Source code for SMS receiver

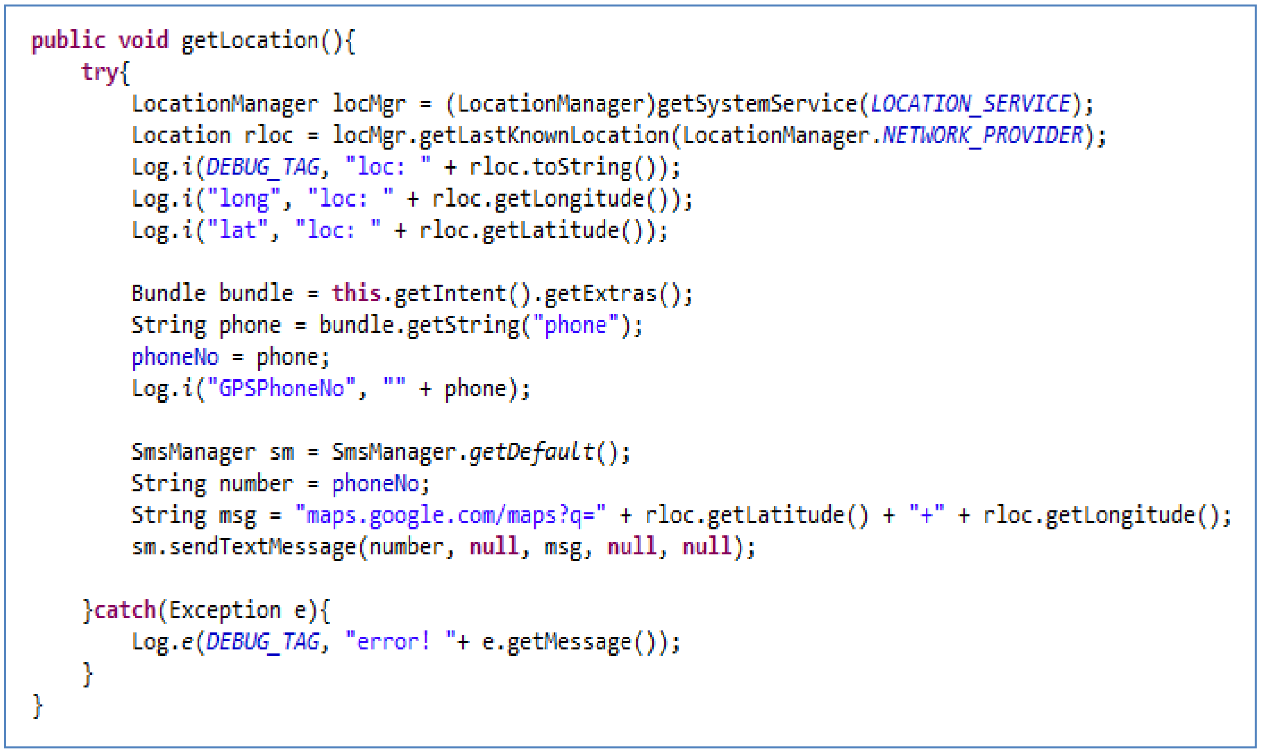
This SMSReceiver class is used to listen for incoming message. It will read the message that is received from the message inbox.



```
// show sender phone number
Toast toast1 = Toast.makeText(context, "Sender: "
        + smsMessage[0].getOriginatingAddress(), Toast.LENGTH_LONG);
toast1.show();

// show message
Toast toast = Toast.makeText(context, "Received SMS: "
        + smsMessage[0].getMessageBody(), Toast.LENGTH_LONG);
toast.show();
```

Figure 4.1.11 : Source code for showing SMS sender and message

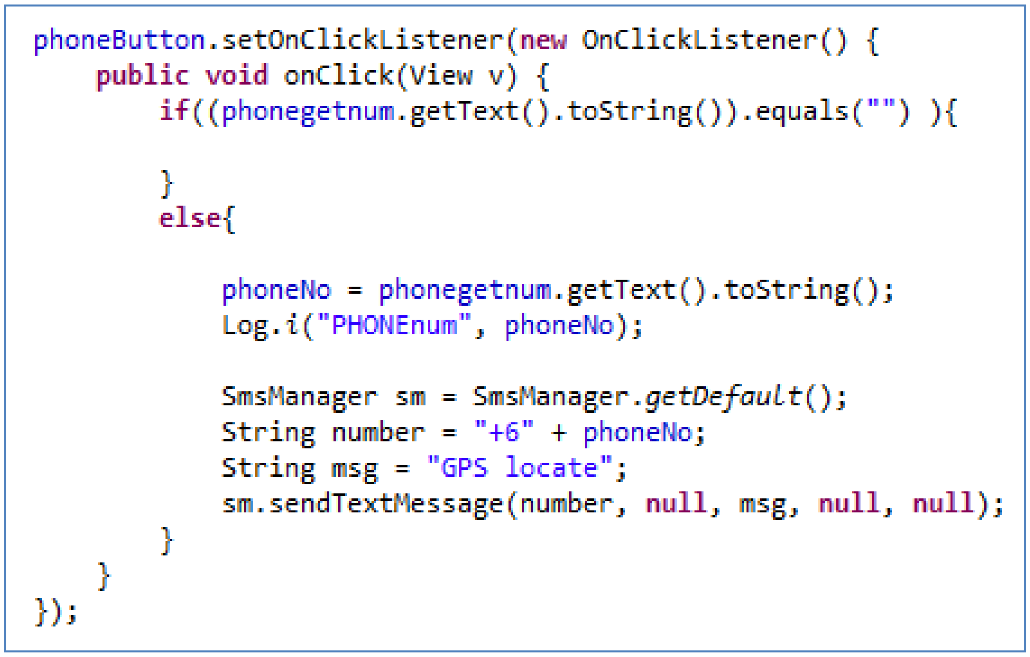
This part of code will display to the user who is the sender by showing the sender's phone number and also will show the message that is received.



```
public void getLocation(){
    try{
        LocationManager locMgr = (LocationManager)getSystemService(LOCATION_SERVICE);
        Location rloc = locMgr.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
        Log.i(DEBUG_TAG, "loc: " + rloc.toString());
        Log.i("long", "loc: " + rloc.getLongitude());
        Log.i("lat", "loc: " + rloc.getLatitude());

        Bundle bundle = this.getIntent().getExtras();
        String phone = bundle.getString("phone");
        phoneNo = phone;
        Log.i("GPSPhoneNo", "" + phone);

        SmsManager sm = SmsManager.getDefault();
        String number = phoneNo;
        String msg = "maps.google.com/maps?q=" + rloc.getLatitude() + "+" + rloc.getLongitude();
        sm.sendTextMessage(number, null, msg, null, null);

    }catch(Exception e){
        Log.e(DEBUG_TAG, "error! "+ e.getMessage());
    }
}
```

Figure 4.1.12 : Source code for geoLocation

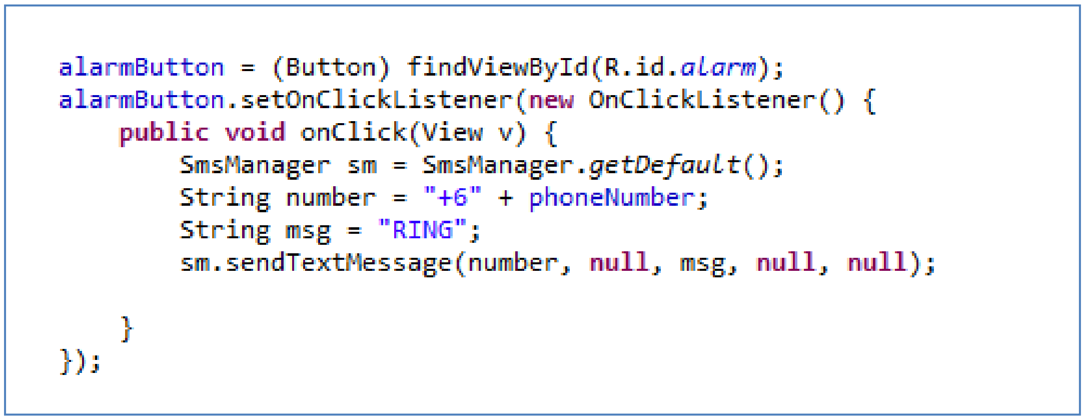
This getLocation() class is to get the latitude and the longitude of the lost device. So when a device is lost, the user can retrieve back the location of the lost device by simply sending over a message to the lost device which will trigger this class to send back the location link of the lost device automatically.



```
phoneButton.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        if((phonegetnum.getText().toString()).equals("") ){

        }
        else{

            phoneNo = phonegetnum.getText().toString();
            Log.i("PHONEnum", phoneNo);

            SmsManager sm = SmsManager.getDefault();
            String number = "+6" + phoneNo;
            String msg = "GPS locate";
            sm.sendTextMessage(number, null, msg, null, null);
        }
    }
});
```

Figure 4.1.13 : Source code for GPS message sender

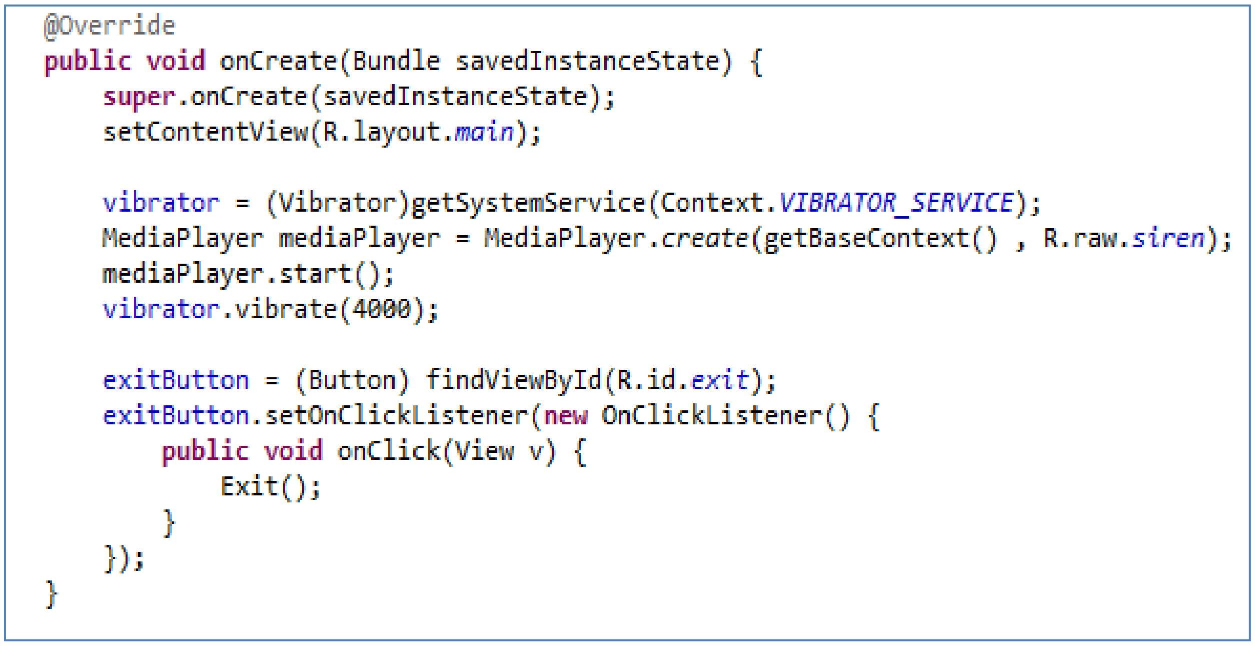
This phoneButton is add-on to allow user to personally key in the phone number of the lost device to retrieve back the location link of the lost device. So any phone which is installed with this GeoLocation application will be able to help locate back the location of the lost device.



```
alarmButton = (Button) findViewById(R.id.alarm);
alarmButton.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        SmsManager sm = SmsManager.getDefault();
        String number = "+6" + phoneNumber;
        String msg = "RING";
        sm.sendTextMessage(number, null, msg, null, null);

    }
});
```

Figure 4.1.14 : Source code for alarm button

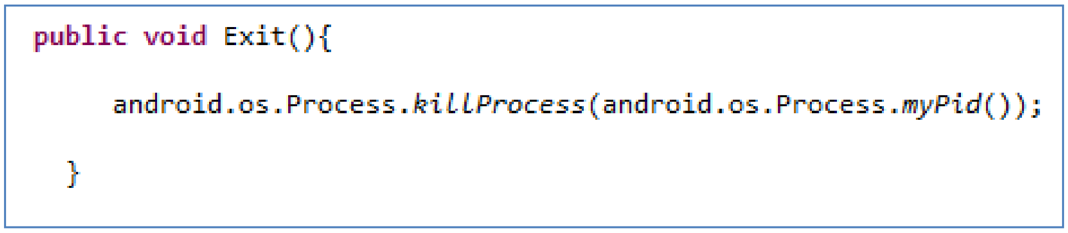
The alarmButton is set in such a way that once the user press the alarm button, the device will send over a unique message to trigger the other phone’s alarm.



```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    vibrator = (Vibrator)getSystemService(Context.VIBRATOR_SERVICE);
    MediaPlayer mediaPlayer = MediaPlayer.create(getBaseContext() , R.raw.siren);
    mediaPlayer.start();
    vibrator.vibrate(4000);

    exitButton = (Button) findViewById(R.id.exit);
    exitButton.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
            Exit();
        }
    });
}
```

Figure 4.1.15 : Source code for alarm

This function is set to sound the alarm whenever the user press on the alarm button above. The alarm will enable the lost device to make noise and vibrate at the same time letting the user know where the lost device is located.

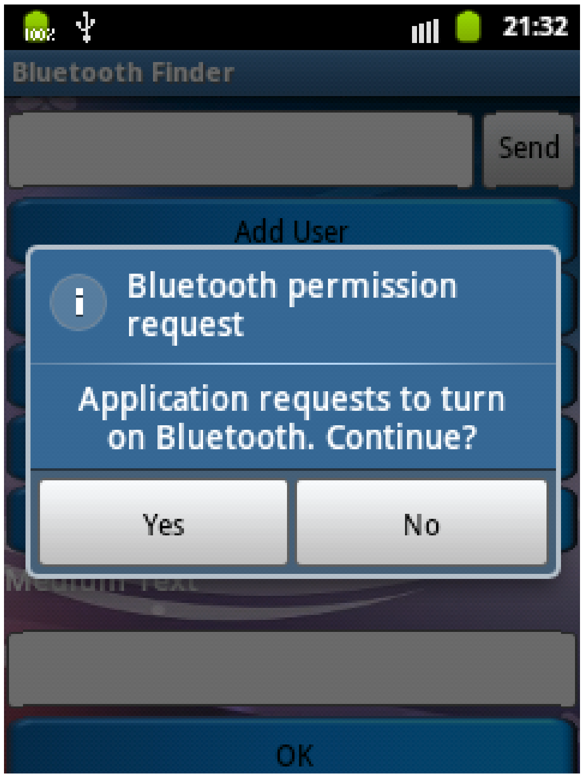


```
public void Exit(){

     android.os.Process.killProcess(android.os.Process.myPid());

  }
```

Figure 4.1.16 : Source code for exit application

This exit function is applied to allow the user to exit the application so that the application no longer runs in the background of the device. This will allow the user to straight kill the application without the need to personally go to the task manager to kill off the application which will continue to run in the background of the device.

4.2 System Interface

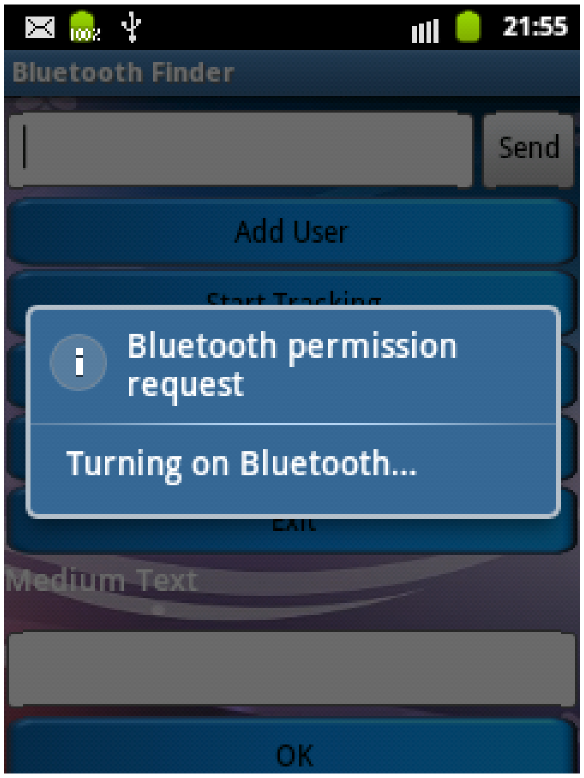


Bluetooth permission request

Application requests to turn on Bluetooth. Continue?

Figure 4.2.1 : Bluetooth Permission Request

Figure 4.2.1 shows the interface where user is prompted by the application to switch on the device Bluetooth. If user presses No, the application will exit. If user presses yes, then user will be directed to the next prompt up which is shown in Figure 4.2.1.

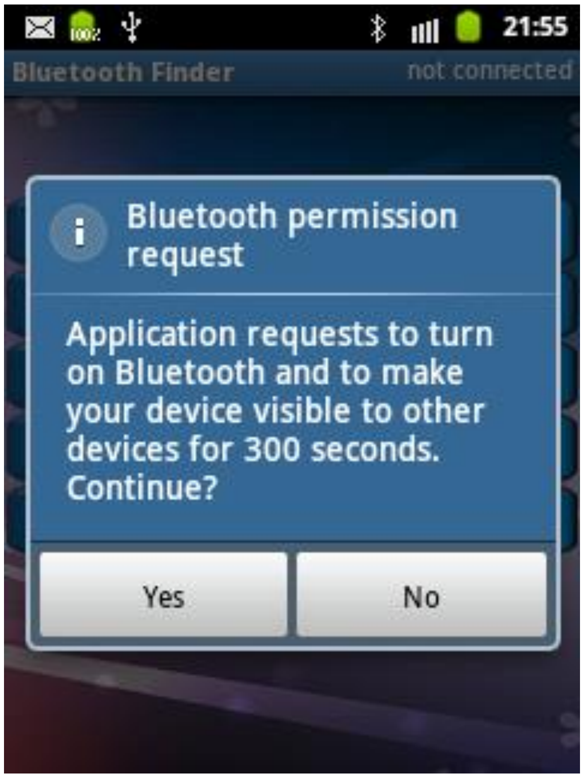


Bluetooth permission request

Turning on Bluetooth...

Figure 4.2.2 : Bluetooth Turning On

In Figure 4.2.2, user is shown the dialog turning on Bluetooth. The application will take this time to turn on the Bluetooth in the user's device before closing this dialog.



Bluetooth permission request

Application requests to turn on Bluetooth and to make your device visible to other devices for 300 seconds. Continue?

Figure 4.2.3 : Bluetooth Permission to Make Discoverable

Figure 4.2.3 is to ask permission from user to set the device to discoverable mode in order for other device to detect the Bluetooth signal from the user's device. If the device is not set to discoverable or visible, other devices will not be able to detect the Bluetooth signal.

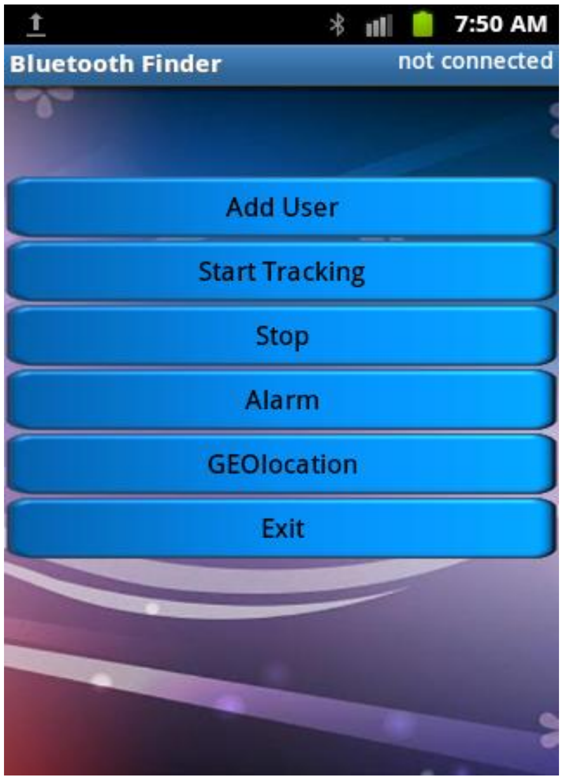
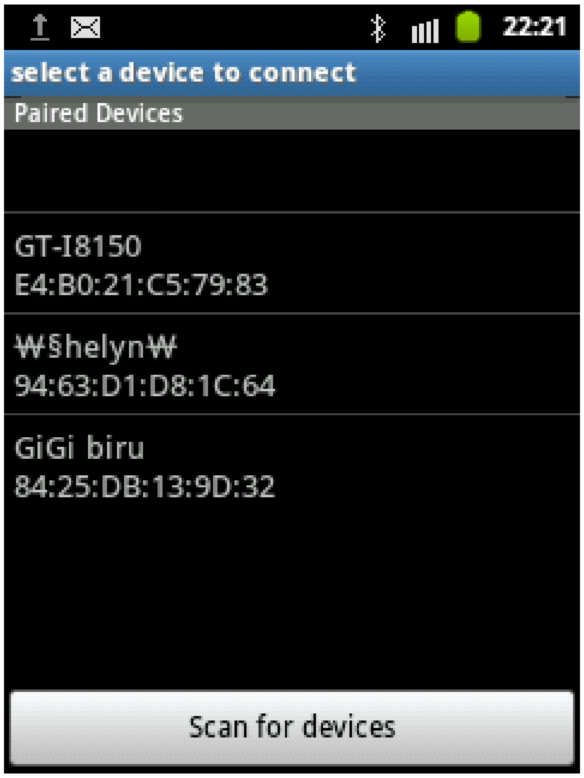


Figure 4.2.4 : Bluetooth Finder Main Menu

Figure 4.2.4 is the main menu for our Bluetooth Finder application. There are few buttons that is added to allow user to easily use the application with ease. The user will first need to add user to start using this application. When the add user button is pressed, Figure 4.2.5 will be prompted up.



22:21

select a device to connect

Paired Devices

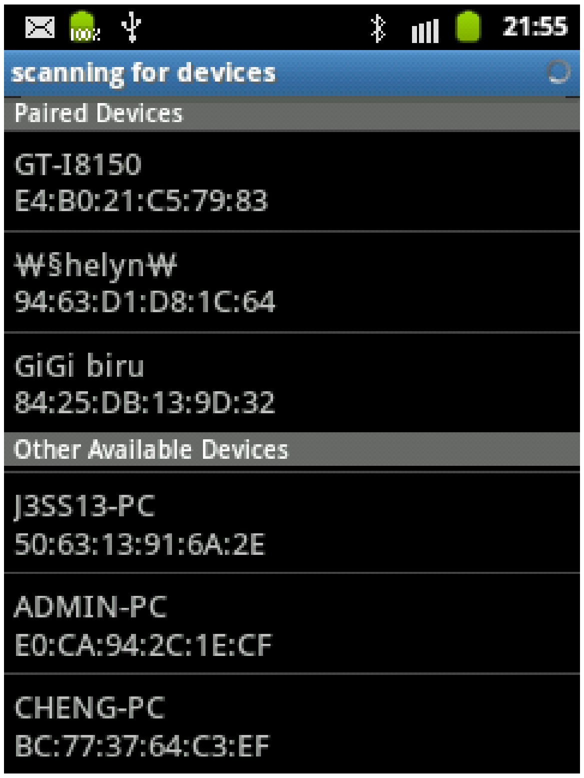
GT-I8150
E4:B0:21:C5:79:83

₩§helyn₩
94:63:D1:D8:1C:64

GiGi biru
84:25:DB:13:9D:32

Figure 4.2.5 : Bluetooth Paired Device Menu

Figure 4.2.5 will be prompted to user to allow user to choose the device that user want to pair with. The device that had been paired before will be listed in the list view. In order for user to search for new device, user will need to press the "Scan for Devices" button which will load the Figure 4.2.6 below.



21:55

scanning for devices

Paired Devices

GT-I8150
E4:B0:21:C5:79:83

₩§helyn₩
94:63:D1:D8:1C:64

GiGi biru
84:25:DB:13:9D:32

Other Available Devices

J3SS13-PC
50:63:13:91:6A:2E

ADMIN-PC
E0:CA:94:2C:1E:CF

CHENG-PC
BC:77:37:64:C3:EF

Figure 4.2.6 : Bluetooth Scanning for Device

Figure 4.2.6 will show all the devices available for pairing with the user's device. Other available device list view will show the devices around with the Bluetooth functionality turn on.

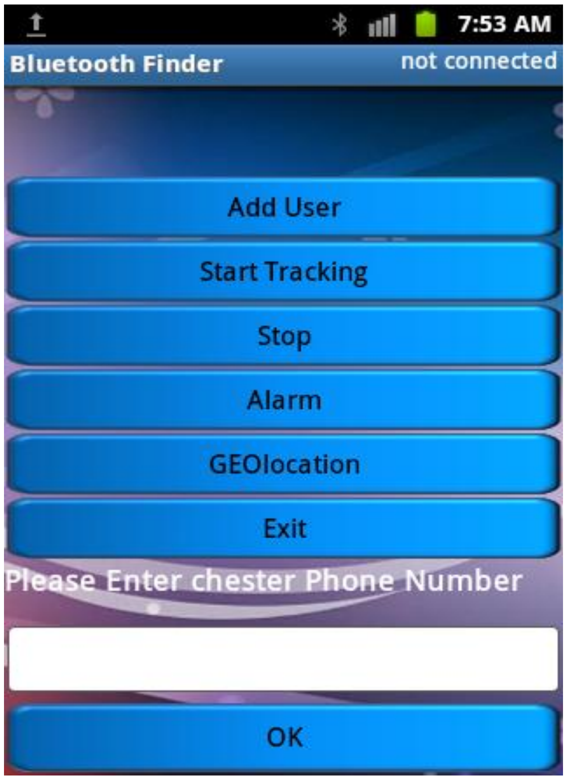
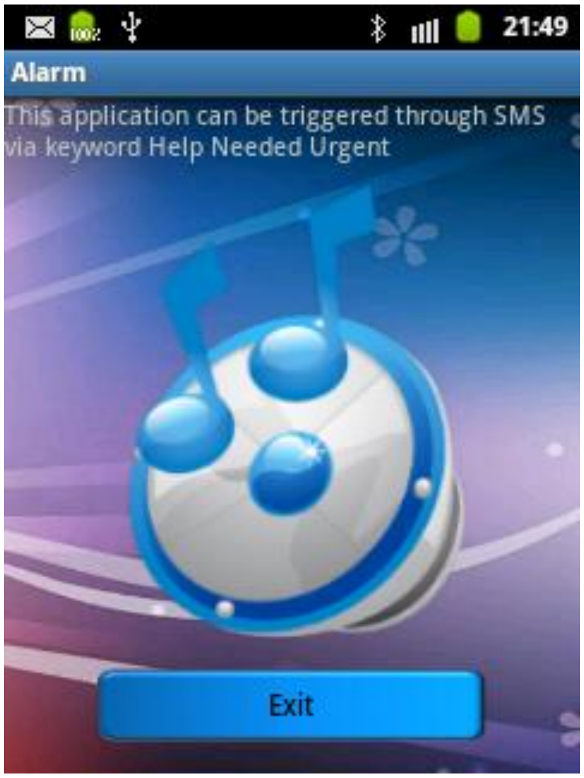


Figure 4.2.7 : Bluetooth Finder Insert Phone Number

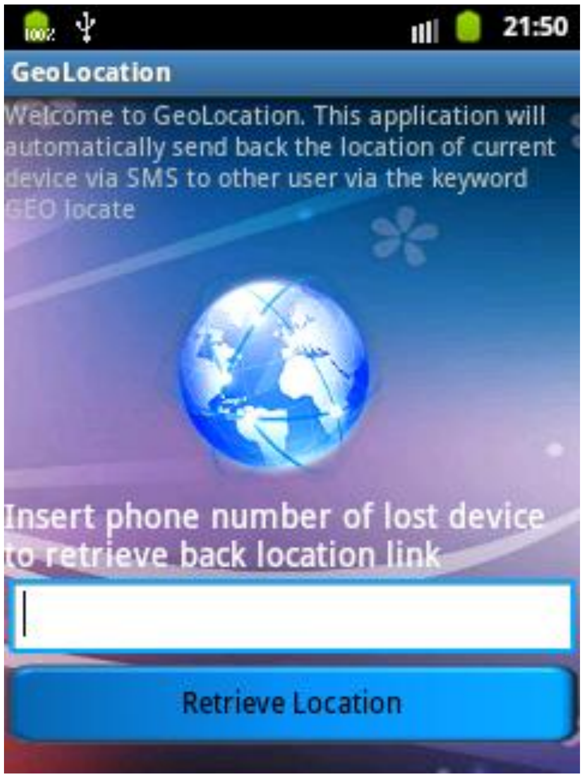
Figure 4.2.7 will be prompted to allow user to key in the phone number of the pairing device once user had already chosen the device from the list view above.



This application can be triggered through SMS via keyword Help Needed Urgent

Figure 4.2.8 : Alarm Menu

Once user hit the alarm button from the Bluetooth Finder application menu, it will automatically trigger the Alarm application of the pairing device. There is also an exit button for user to exit the application.

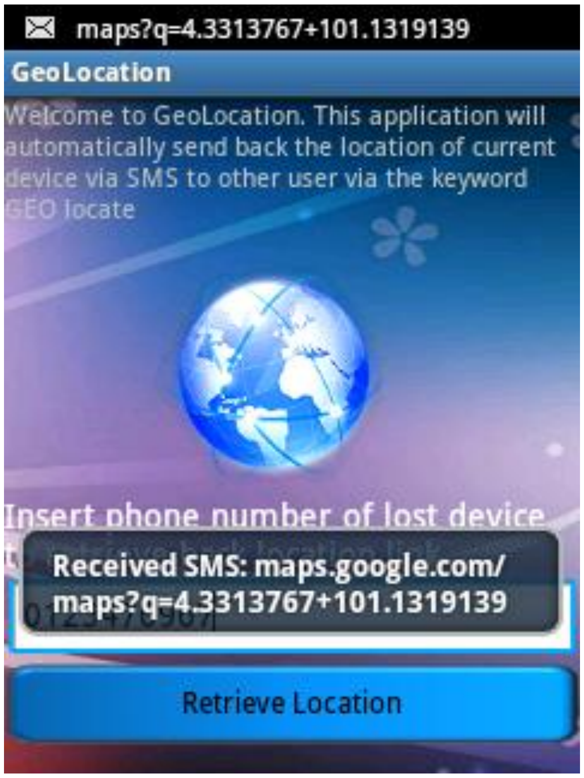


Welcome to GeoLocation. This application will automatically send back the location of current device via SMS to other user via the keyword GEO locate

Insert phone number of lost device to retrieve back location link

Figure 4.2.9 : GeoLocation Menu

As for Figure 4.2.9, this is the GeoLocation application menu. This application can be triggered by user by sending over the key word message “GEO locate” and this application will automatically send over to the sender the location link of the other device.


maps?q=4.3313767+101.1319139

GeoLocation

Welcome to GeoLocation. This application will automatically send back the location of current device via SMS to other user via the keyword GEO locate

Insert phone number of lost device

Received SMS: maps.google.com/maps?q=4.3313767+101.1319139

Figure 4.2.10 : Geolocation SMS Received

Figure 4.2.10 shows the location link being sent over to the sender's device. User can use this location link to find back the lost device by clicking on the link which will activate the google map.

5.0 Testing

System testing is the process of performing a variety of test on a system to explore the functionality of the system and also at the same time to identify what are the problems or errors of the system. We will be conducting several testing on our system to try to find out the main problems and fix it. Besides that, system testing also enable us to test out how far is the distance of the Bluetooth connection which will be explain further on below. We also conduct out a few test cases to find out how our system will works on different circumstances. For example, when we finish up a new function we will test it out to see whether the function works like what we intend it to do. If not then we will continue on debugging and testing again and again. For this project we will be testing it out on our Android mobile devices which are Samsung Galaxy Ace and Samsung Galaxy Y.

5.1 Test Results

Application Features	**Results**	**Testing done by**
Able to establish connection between devices.	✓	Lee Yen Nee
Able to connect more than 1 device.	✓	Lee Yen Nee
Master Side		
Slave device able to keep track of Master device which is disconnected.	✗	Lee Yen Nee
Slave device able to keep track of Master device which is out of range.	✗	Lee Yen Nee
Master and Slave device alarm ring when Master device is out of range.	✗	Lee Yen Nee
Master and Slave device alarm ring when Master device is disconnected.	✗	Lee Yen Nee
Master and Slave device vibrate when Master device is out of range.	✗	Lee Yen Nee

Master and Slave device vibrate when Master device is disconnected.	✗	Lee Yen Nee
Master and Slave device receive SMS when Master device is out of range.	✗	Lee Yen Nee
Master and Slave device receive SMS when Master device is disconnected.	✗	Lee Yen Nee
Master device reconnect back to Slave device when Master device is nearby.	✓	Lim Yuen Nee
Master device send SMS to get back location of Slave device.	✓	Lim Yuen Nee
Master device receive location SMS from Slave device.	✓	Lim Yuen Nee
Master device sound the alarm of Slave device when Slave device is nearby.	✓	Lim Yuen Nee
Slave side		
Master device able to keep track of Slave device which is disconnected.	✓	Lee Yen Nee
Master device able to keep track of Slave device which is out of range.	✓	Lee Yen Nee
Master and Slave device alarm ring when Slave device is out of range.	✓	Lee Yen Nee
Master and Slave device alarm ring when Slave device is disconnected.	✓	Lee Yen Nee
Master and Slave device vibrate when Slave device is out of range.	✓	Lee Yen Nee
Master and Slave device vibrate when Slave device is disconnected.	✓	Lee Yen Nee
Master and Slave device receive SMS when Slave device is out of range.	✓	Lee Yen Nee

Master and Slave device receive SMS when Slave device is disconnected.	✓	Lee Yen Nee
Master device receive SMS when Slave device shake the phone.	✓	Lee Yen Nee
Slave device receive SMS when Master device shake the phone.	✓	Lee Yen Nee
Master device receive SMS when Slave device battery is low.	✓	Lee Yen Nee
Slave device receive SMS when Master device battery is low.	✓	Lee Yen Nee
Slave device reconnect back to Master device when Slave device is nearby.	✓	Lim Yuen Nee
Slave device send SMS to get back location of Master device.	✓	Lim Yuen Nee
Slave device receive location SMS from Master device.	✓	Lim Yuen Nee
Slave device sound the alarm of Master device when Master device is nearby.	✓	Lim Yuen Nee

Table 5.1.1 : Test Cases

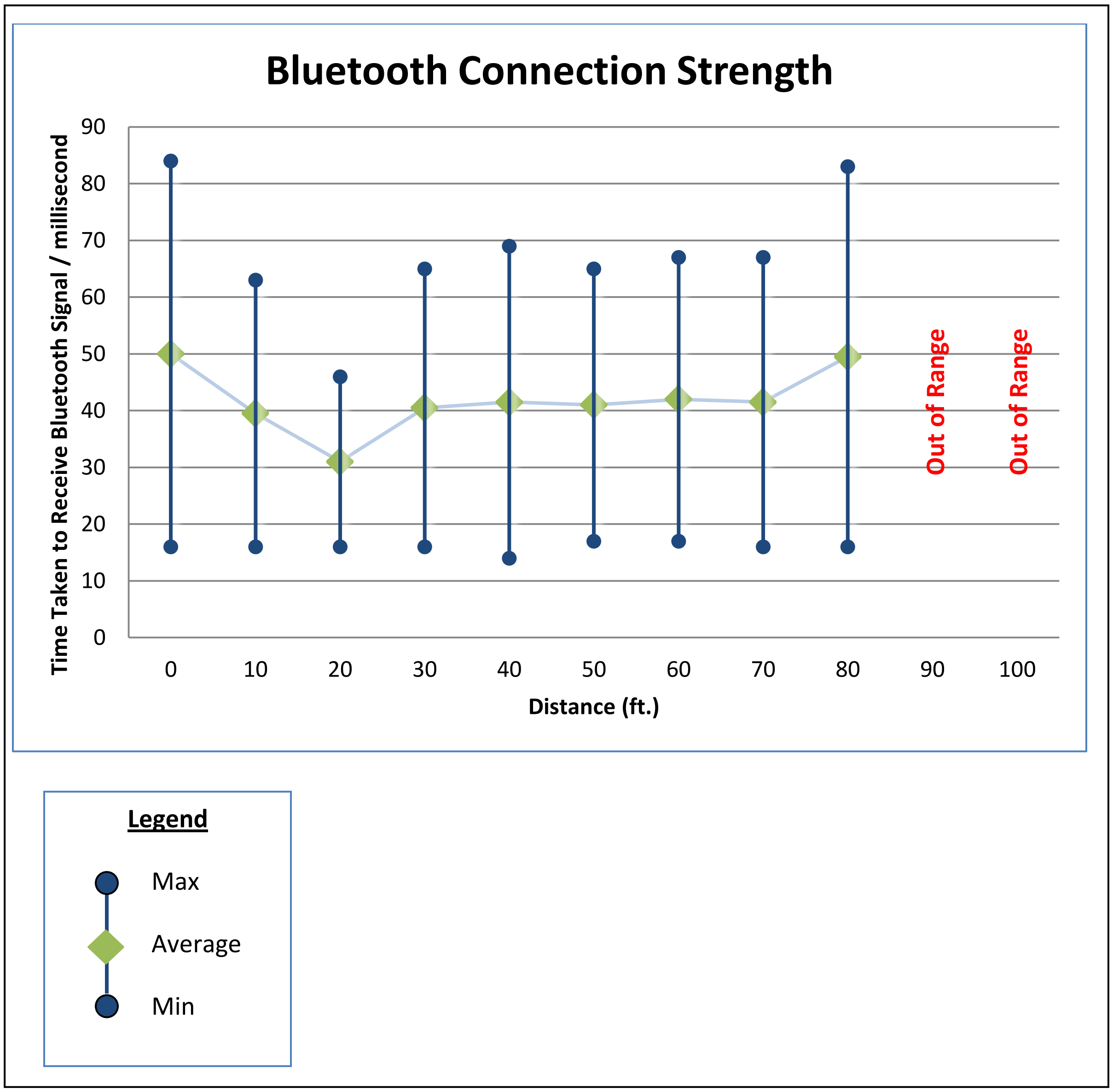


Figure 5.1.1 : Graph on Bluetooth Connection Strength

Based on Figure 5.1.1, it shows the experimental results of the testing process that we had carried out to compare on the Bluetooth signal strength. This experiment is carried out using Android devices and in an open area where there are nothing interfering with the Bluetooth signal. The first purpose for this test is to measure how far the Bluetooth device can connect with each other without going out of range. As we can see from Figure 5.1.1, once the distance between the two devices goes above 80ft, the device will immediately lose its Bluetooth signal. The second purpose is to measure how long it will take for the device to transmit a string with the size of 512Kb to another device without losing its Bluetooth signal. We found that this second experiment that we carried out is not feasible as the average results that we had gotten are around 30 to 50 milliseconds. So overall, the conclusion that we can make from this experiment is that Bluetooth device will disconnect when the device goes beyond 80ft in distance.

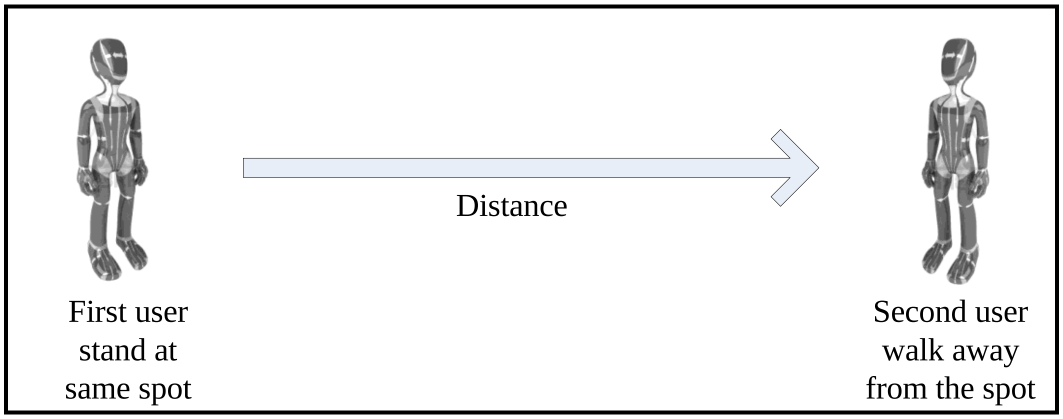

Distance
First user stand at same spot
Second user walk away from the spot

Figure 5.1.2 : How testing is carried out

Our testing is carried out by having two people holding on to an Android mobile device. One person will stand still at a certain spot while another person will slowly walk away from the spot in a straight line without having anything interfering the signal in between them. The distance is calculated by feet. At every 10feet, the second user will send a message back to the first user and the first user mobile device will automatically process the message and send back the message saying that it had receive the message. The second user device will also keep track of how long it will take for the message to be sent out and receive back in milliseconds. Then the second user will continue on this flow until the Bluetooth connection is disconnected.

6.0 Conclusion

There are lots of similar applications in the market nowadays. By doing this project, it will be able to help those users who are carrying around many belongings and at the same time the need to feel secure that their belongings don’t get too far away from them. It can also reduce down the search area scope by implementing the GeoLocation and also the alarm functionality. So basically our main objective was to be able to auto-connect to the lost pairing device, to track back the lost Bluetooth device and also to locate the lost Bluetooth device. The first objective can be done by implementing the auto-connection function which the device will keep on trying to reconnect back the lost device, so as long as the lost device gets nearby, the device will automatically reconnect back to the user’s device. While the second objective can be achieve by applying the Geolocation functionality. This Geolocation functionality can be trigger by user by sending over a special message to the lost device to trigger the lost device GeoLocation functionality to automatically send back the location link of its own self to the user. So user can use this location link to search for the lost device. The third objective can be achieved by using the alarm button to locate back the lost device. Once user knows the location of the lost device and get near to the lost device, the user can trigger the alarm of the lost device so that the lost device will make some noise to alert the user. So overall, all the objective of this project is achievable.

6.1 Limitation

This project is to create a Bluetooth application to search back on the lost Bluetooth. However, there are some limitations in this application. The first limitation is when device is out of credit. This will cause the user not being able to send out SMS to retrieve back the lost device location. If the lost device is also out of credit then it will also not being able to automatically send back its current location to the user who requested for it.

The next limitation is the alarm function in this application is triggered by SMS as well. This is because the Bluetooth connection will keep on pairing when connection is made. So this will cause the application not being unable to process other things at the same time.

The third limitation is that the GeoLocation functionality could not produce out the same results on different model of phones. When used in different phones it will force close causing the whole function not workable. Besides that, this function also takes some time to send back the current location of the lost device on the first attempt.

6.2 Future Work

This application can be better improve or enhance by making sure that both user have some credit left before Bluetooth connection is allowed to be made. So this will ensure that when device got lost the SMS will go through and not stuck there due to insufficient credit in the mobile phone.

Next this application can also try to implement in the alarm function via Bluetooth connection. This is to safe up on the credit and since we are already doing this Bluetooth connection. User can easily send over a help message via Bluetooth connection and this will help to reduce the need send a help message just to trigger the alarm.

The application can also be improved by getting the GeoLocation functionality more stable meaning that it can runs smoothly on different mobile devices without having to force close. This is to allow user to search back on the lost devices more efficient and faster.

7.0 References

AppBrain, *Welcome to the AppBrain Android market*. Available from: <http://www.appbrain.com/>. [28 Feb 2012].

Atcle, 2011, *Keep bluetooth from lost*. Available from: <http://www.appbrain.com/app/keep-bluetooth-from-lost/com.atcle>. [28 Feb 2012].

Bluetooth SIG, Bluetooth Basics. Available from: <http://www.bluetooth.com/Pages/Basics.aspx>. [2 March 2012].

Bluetooth SIG, *Fast Fact*. Available from: <http://www.bluetooth.com/Pages/Fast-Facts.aspx>. [2 March 2012].

Bluetooth Tomorrow, *Bluetooth Advantages - Why Use Bluetooth?.* Available from: <http://www.bluetomorrow.com/about-bluetooth-technology/general-bluetooth information/ bluetooth-advantages.html>. [2 March 2012].

Bowl, P, 2011, *Bluetooth Tracker.* Available from: <http://www.appbrain.com/app/bluetooth-tracker/com.purplebowl.BluetoothTracker>. [28 Feb 2012].

Build Security In, 2009, *White Box Testing*. Available from: <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/white-box/259-BSI.html>. [1 April 2012].

Chris, 2009, *GPS vs. aGPS: A Quick Tutorial*. Available from: <http://www.wpcentral.com/gps-vs-agps-quick-tutorial>. [2 March 2012].

David.B, 2012, *Spiral model*. Available from: <http://en.wikipedia.org/wiki/Spiral_model>. [2 March 2012]

L.C.José, 2011, *Bluetooth Finder*. Available from: <http://www.appbrain.com/app/bluetooth-finder/com.bluetoothFinder>. [28 Feb 2012].

Laurie.W, 2006, *Testing Overview and Black-Box Testing Techniques*. Available from: <http://agile.csc.ncsu.edu/SEMaterials/BlackBox.pdf>. [1 April 2012].

Laurie.W, 2006, *White-Box Testing*. Available from: <http://agile.csc.ncsu.edu/SEMaterials/WhiteBox.pdf>. [1 April 2012].

Machine Design, *Proximity Sensor*. Available from: <http://www.sensorstransducers. machinedesign.com/guiEdits/Content/bdeee4/bdeee4_7.aspx>. [2 March 2012].

Our Wireless World, 2011, *Baby Bluetooth Alarm*. Available from: <http://www.appbrain.com/app/baby-bluetooth-alarm/com.app.babybluetooth#>. [28 Feb 2012].

Software Testing Help, *Black Box Testing: Types and techniques of BBT*. Available from: <http://www.softwaretestinghelp.com/black-box-testing/>. [1 April 2012].

Software Testing Help, *White box testing: Need, Skill required and Limitations*. Available from: <http://www.softwaretestinghelp.com/white-box-testing/>. [1 April 2012].

SSPARK, 2011, *Bluetooth Alarm*. Available from: <http://www.appbrain.com/app/bluetooth-alarm/jp.sspark.bluetooth>. [28 Feb 2012].